

Running Cipher Contests

Arup Guha

August 5, 2004

COT 5937 – Introduction to Cryptography

- Proposed the course based on two cryptography courses I took in college
- Taught the course three times
 - Summer 2001
 - Summer 2003 (with my dad)
 - Summer 2004
- My interest: Primarily Mathematical

Practical Application of Algorithm Analysis

- Writing your own BigInt class including modular exponentiation
 - Iterative exponentiation is incredibly slow.
- Using Shanks Algorithm to solve the Discrete Log problem
 - Sorting 1,000,000 using an Insertion Sort doesn't cut it
 - Some BigInteger methods in Java are time intensive.
- ModPow
 - Incorrect use (first doing pow), then mod at the end is also incredibly slow!!!

Why a contest?

- While it was fun to break the professor's homework assignments, wouldn't it be more fun to break a classmate's code?
- More creative than the cryptanalysis typically given in a homework assignment
- Breaking codes has been historically significant; this can get students excited!

2001: Contest Parameters

- Students were introduced to substitution, Vigenere and Transposition before they had to come up with their code.
- Key size: 2^{25}
- Product cipher of two simple ciphers (eliminating multiple rounds)

2001: Contest Parameters cont.

- Each team was paired against another team based on an in-class contest.
- Each team was given three ciphertexts encrypted with the opponent's code and key. (The opponent chose the key.)
- At a later date, each team received matching plain and cipher texts.

2001: Contest Parameters cont.

- Each team got the opportunity to receive ciphertext for a chosen plaintext.
- Two weeks before the end of the contest, each team received an algorithmic description of the opponent's code.
- To help, I encrypted part of a message with my encryption scheme. If they could break mine, it would give them information about one of their ciphertexts.

Contest Results

- Nearly all teams were able to break their respective codes once they had the algorithm to use for two weeks.
- The winning team was able to determine a pattern that helped them set up a set of equations to solve for part of the key.
- Every team spent a good deal of time with basic cryptanalysis before they received the algorithm that didn't get them anywhere.

Contest Results

- Some team's cipher's were considerably stronger than others, so some groups' tasks were a lot easier than others.
- The contest is time-consuming for teams and does take away some time from other class assignments.
- All the teams had fun trying to crack codes.

2003 Contest Changes

- Key size = 2^{32}
- Teams did not received all information at the same time.
 - In class contests were run, and the winners received matching plain and ciphertext first.
 - Winners received ciphertext for chosen plaintext based on a contest.
 - Finally, winners received algorithmic descriptions based on an in-class contest.

In-class Contest

- A question based on the previous lecture
- First team to answer correctly wins.
- An incorrect response disqualifies you. (So it's important to be correct.)
- This integrates the contest with the class material
- Drawback – Teams with “faster” students usually win, thereby giving them an advantage.

Results of the in-class Contest

- Teams were encouraged to keep up with class material in the hopes of winning information before their counterpart.
- The in-class contest certainly affected the overall contest since teams had information for an unequal amount of time.

Results of 2003 Contest

- The winning team won without even the chosen plaintext. All they used was the matching plaintext.
 - Their opponent's cipher was very weak since they didn't spend a whole lot of time on the assignment.
 - It was difficult to tell how the winners would have fared against a much stronger cipher.

Results of 2003 Contest cont.

- All teams except one broke the opposing cipher.
 - The problem with that one team is that the description they received didn't match the actual code used for encryption.
 - The other teams did use some cryptanalysis to break the original messages.

2004 Contest Changes

- Key size: 2^{48}
- No restriction on the complexity of the cipher (so multiple rounds allowed)
- All teams received the exact code for encryption towards the end of the contest (not decryption though).
- The original ciphertexts actually gave directions to a buried “treasure.”
- I picked the keys for each cipher system.

Results of 2004 Contest

- Teams were able to do very little without the code.
 - All teams sufficiently removed frequency information to render the basic cryptanalysis techniques moot.
 - The length of the ciphertexts varied greatly since restrictions weren't placed on the number of rounds

Results of 2004 Contest cont.

- The in-class contest to win the code (the last stage) was incredibly important.
 - One of the team's that won the opposing team's encryption code was able to win the contest before the next class meeting.
 - But, the other team that won the opposing team's encryption code was unable to break the ciphertexts at all.

Results of 2004 Contest cont.

- Each cryptosystem (except for one) was much, much more complicated than the ones in year's past
- Many teams used Java's random number generator, which made cryptanalysis more difficult.
- At first, due to the complexity, I couldn't get several team's decryption code to reverse encryption.

Results of 2004 Contest cont.

- There was more opportunity for cryptanalysis with the code
 - The winner set up a system of 27 equations based on a plaintext/ciphertext matching pair and solved for the key uniquely!
 - Another team recovered 25 bits of the key through the cryptanalysis of only the last two stages of the opponent's cipher and used a brute force search for the remaining 23 bits.

Results of 2004 Contest cont.

- A third team noticed that the ratio of the length of the plaintext to the ciphertext varied dependant on the first 8 bits of the key. They used this ratio to solve for the first 8 bits of the key. They solved for the last five bits of the key as well noticing that the same digraph always ended up in a particular location of all ciphertexts. They then tried all 32 possibilities for the last five bits and found that only one of those could have created the digraph they saw.

Results of 2004 Contest cont.

- A fourth team was able to break the code they received in less than an hour by setting up equations related to a stream vigenere cipher that let them also solve directly for the key
- Only three teams (out of eight) broke their respective codes
 - One cipher was easy to break
 - Two ciphers were difficult, but the teams came up with clever cryptanalysis to find the key or reduce the key space.

Results of 2004 Contest cont.

- Five teams were unable to break their codes.
 - One team mentioned reduced the key space to 2^{35} , but couldn't do a brute force search due to how long the cipher took to execute once.
 - A couple teams got stuck trying to reverse engineer Java's Random class.
 - Only one team didn't put forth a sufficient effort to break the code they were assigned.

Changes I'd Make

- Give out the code earlier.
 - This way teams waste less time going down completely wrong paths.
- Give teams longer to develop their ciphers.
 - I can check on their progress so that their final submission actually works
- Restrict the key format.
- Restrict the length of the ciphertext based on the plaintext length.