Cody Biggo, M.Sc. Eng.
Quality Assurance (QA) Manager
Cryptorama.COM
U Be St.
Deco, DE 3117

April 10, 2013

Discreet Mathematicians Consulting
600 W. Walnut Street
Danville, KY 40244

Project Groups:
(Grant, Blaise),
(Emily, Hannah),
(Cyrus, Frances),
(Anne, Forrest),
(Peter, Michael, Henry),
(Cole, Yuchen),
(Levi, Chase),
(Woody, Matt),
(Simon, Liz),
(Jess, Lindsey)

Dear Discreet Math Consultants:

I am the quality assurance (QA) manager in the "New Codes" department of Cryptorama.COM. On the weekends I play WoW with Phil Bonacci (a.k.a. Nano-Leader) in the Angry Armadillo guild, and it was Phil who recommended your services to me. He mentioned that you had rescued his company from a very sticky situation (he didn't tell me exactly what it was – apparently he does some kind of classified work?). Anyway, I have an embarrassing problem on my hands, and I'm hoping you can help me out too.

One group of cryptographers (Team Azure) here at Cryptorama has developed a new code, called the "SuperLooper Cipher". Similar to the age-old Caesar cipher, it's a *symmetric key* algorithm – i.e., the algorithm for decoding uses the same key as the algorithm for encoding. (Not to be confused with public key algorithms like RSA... but I digress.) Now, the Caesar cipher is known to be a very weak encoding mechanism, but Team Azure claims that SuperLooper is much more secure, and defeats letter frequency counting by using the offset in the message as part of the encoding scheme – thus, the same letter in the plaintext won't always get mapped to the same letter in the ciphertext. However, Team Blitzen is skeptical about the algorithm, and is quite sure it must have some serious flaws (even though they haven't positively identified any flaws themselves).

The truly embarrassing part is that my whole division has turned into a nasty name-calling epithet-shouting Nerf-gun wielding free-for-all --- one cryptographer even threatened to sic his pet Gila monster on the other team if they didn't admit they were wrong. So productivity in the office has dropped to zero (perhaps negative 10?) and I'm stuck in the middle trying to keep the peace. Both teams are too emotionally involved in this debate to think clearly or logically at this point, so I need an unbiased outside source to evaluate this algorithm and settle the matter.

I have used a secure messaging protocol to transmit the code for the SuperLooper Cipher algorithm to your consulting firm's coordinator, Dr. Stonedahl, and have asked him to make it available to you in digital form. The code was written using Python (version 2.6), and should run on any system with Python installed, without any external libraries. You may need to run it and/or edit it in order to complete the tasks I am requesting.

Here are the key points that need to be resolved:
1) How secure is the algorithm really? If the people who intercept the message don't know the encryption key, (but do know our code/algorithm) how hard would it be for them to decode it? I

am including two encrypted messages (enciphered using two different unsigned 64-bit keys – integers between 0 and 2^{64}-1). I want to see if you can decode either of them.

Sample Cipher-Text 1: **`TEF DWE UGSWE GCVCMI NWSX! XLNBDR!`**
Sample Cipher-Text 2: **`DVJ RI EQJZOQ PN PKO ZNS BHR IRBX XGMYGYS?`**

2) Team Azure claims that since there are 2^{64} possible keys, the algorithm is immune to "brute force" attacks, where the attacker tries all possible keys. I agree with them that it would take way too long for an attacker to check that many keys. However is it possible that the attacker could decode the message by trying only some modest subset of the keys? (Team Blitzen expressed doubts about whether every key corresponds to a unique encoding.)

3) Are some choices of keys particularly bad or particularly good?

4) For some larger messages and/or larger keys, the encoding/decoding functions seems to run rather slowly. Could you have your algorithm analysis experts analyze the algorithm's efficiency? How many times will the "modulus" (%) operator be run (in the worst case), expressed as a function of **two** variables: N - the number of alphabetic characters in the plaintext message, and K - the encryption key. What is the "order" of this function (expressed in O/Ω/Θ notation), again expressed as a function both variables?

5) Provide a table AND a graph showing the actual amount of time (in seconds) that the algorithm takes to encode plaintext messages of varying sizes: N = 10,20,30,...,100 assuming a fixed value of K=10000. (There's some example timing code in the file, and it's easy to find other methods using Google.) Does your timing data agree with your theoretical analysis?

6) Provide another table AND a graph showing the actual amount of time (in seconds) that the algorithm takes to encode plaintext messages of varying keys: K = 1000, 2000, 3000, …, 10000 assuming a fixed value of N=100. Does your timing data agree with your theoretical analysis?

7) Can you modify the existing code to make it run *much more* efficiently, while still giving the same results? (I'm not familiar enough with modular arithmetic to tackle this myself.)

8) Would you recommend that I approve this new encryption algorithm, or tell Team Azure to go back to the drawing board? If you think the SuperLooper Cipher is too weak, do you have any suggestions about how the security might be improved?

For all points, please clearly describe your methods of investigation/analysis, so I can provide a thorough justification to my unruly teams of cryptographers. I would appreciate a reply as soon as possible, but certainly no later than **Monday, April 22**, when the company's senior executives will be reviewing my department.

Respectfully yours,

Cody Biggo, M.Sc. Eng.
Quality Assurance (QA) Manager
Cryptorama.COM