Independent Mathematical Contractors, Inc.

[redacted]

Group H

136 TMCB
Provo, UT 84602

September 9, 2015

OCRAI
OCRA Creative Recursive Acronyms, Inc.
485 Primality Way
Provo, UT 84604

Security Cipher Technical Report

Introduction:

We are technical contractors who have been commissioned by your company to create a nontrivial method of encrypting plaintext. Your company has recently had several major security breaches when sending text messages, so it is our job to make sure that this doesn't happen anymore. After creating this key, your engineers will use our cryptosystem to create a mobile application to encrypt all messages before they are sent. This cryptosystem we created roughly doubles the number of characters. We have created the following method of encrypting plaintext, and believe that it will help protect the sensitive information contained in your text messages.

Solution:

When faced with the task of encrypting your data, we had two distinct ideas. The first idea was to use a cipher that incorporated a system based on the properties of odd and even numbers. We ultimately decided that this method wasn't as secure as our second option, which is the one we decided to use. As we worked on improving security, our basic encryption method stayed the same, but we altered the complexity of the key.

The encryption method that we designed incorporates a table of all the desired letters and symbols. The table that we used includes, in this specific order: letters A-Z (in alphabetical order), space, "." , "," , and "?" . The way that the letters and symbols are inputted into the table is determined by the key given.

A generic example of the table is given below:

	1	2	3	4	5	6
7	A	B	C	D	E	F
8	G	H	I	J	K	L
9	M	N	O	P	Q	R
10	S	T	U	V	W	X
11	Y	Z	space	.	,	?

Each column and row is labeled with a number and it is these numbers that are paired together (column, row) and used to encrypt the letters and symbols of the plaintext. For example, the letter A would be represented as 17. When writing the ciphertext, no spaces are included in between the numbers because spaces are encrypted as well. Using the above table, the phrase "Have a good day." would be 28174105731117311183939473114717111411. A code without spaces increases the strength of the encryption and makes it more difficult to crack without the key.

The key of the code is essential to the security of the encryption method and allows for multiple variations of the encryption table used. An example key is F28LU35216. There are two distinct parts to the key. The first half tells the coder how to organize the encryption table. The first character in the key is the first letter put into the encryption table, and the number that comes next in the key is the table cell to put that letter in. Our example key would assign the letter F to cell 28 in the encryption table.

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30

The two letters that come next decide which direction the letters (in order) will be filled into the table. The first letter can be either R (right) or L (left) which determines the

direction along the row. The L in our example key means that we are moving right to left along the row, and the letter "G" would be put in cell 27, letter "H" would be put in cell 26, etc. The second letter can be U (up) or D (down), which determines how to move from one row to the next. In our example key, the U means that letter "J" would go in cell 24, letter "K" would go in cell 23, etc. Here is how the first half of the key would change our table:

C	B	A	?	,	.
space	…				
			…	K	J
I	H	G	F	E	D

The second half of the key rearranges, or permutes, the columns of the encryption table. The first number in the second half of the key is the column number that is moved into the first column of the encryption table, the second number is moved into the second column and so forth. The column number that is not included in the permutation is placed in the last column. For our example, column 3 will be the first column.

3	5	2	1	6	4
A	,	B	C	.	?
Y	W	Z	space	V	X
S	Q	T	U	P	R
M	K	N	O	J	L
G	E	H	I	D	F

We then change the columns back to the original numbering (1-6) and we now have the final encryption table. Our final table for this encryption key would be:

	1	2	3	4	5	6
7	**A**	**,**	**B**	**C**	**.**	**?**
8	**Y**	**W**	**Z**	**space**	**V**	**X**
9	**S**	**Q**	**T**	**U**	**P**	**R**
10	**M**	**K**	**N**	**O**	**J**	**L**
11	**G**	**E**	**H**	**I**	**D**	**F**

We then use this encryption table to encode the message. Using this new table, the phrase "Have a good day." would be: 31117582114871111410410511485111171857.

In order to decrypt this message, the numbers will have to be read as pairs or triples. A number between 1 and 6 will always come before a number from 7 to 11, which keeps 111 from being interpreted erroneously.

Conclusion:

As can be seen, the technique used to manipulate our data is far from trivial. It involves several permutations of an original chart, and then relies on those permutations to encode (and decode) the text. If even one column is switched erroneously or one number is placed out of order, the entire system will be broken and the individual is unlikely to understand the encoded message being sent. In addition, the fact that each letter has the possibility of being represented by two or three numbers, depending on its position in the table, makes the code stronger. It will be initially unclear if the string of characters represents words, letters or numbers, etc. The increased length of these messages because of the multiple numbers used for each plaintext character will increase the number of possible solutions to our code, and will thereby increase the difficulty of the problem to be solved. At ten characters, the keys are within the bounds of your processing and monetary limits. We believe this will be well worth the added security of this difficult code. We hope that your

team will be satisfied with this and that you will let us know about any feedback or concerns you have at your earliest convenience.