Dear Mr. Andrews,

We at IMC understand the need for secure communication between employees. We understand that your company is looking for a non-trivial method of encrypting and decrypting text messages between employees of your company. The cryptosystem we have created will encrypt a text message in such a way that if it were sent to an unauthorized person, they would not be able to understand nor decrypt the text without the decryption key. Our method is easy to understand and can be done by hand with a little bit of work, so we have included a program with instructions that can do the encryption. This program, with instructions on how to use it, is found at <https://github.com/ThaWeatherman/dbct>.

The algorithm we have developed to secure text messages sent by your company phones is complex enough to resist attacks, but is very easy to use and implement. We used a simplified version of a method known as a block cipher, which we will explain.

Each message that is sent from a company phone will be encrypted by the mobile application. The program first breaks each message into blocks of ten characters. Suppose the first block of ten characters was “abcdefghij.” Each character is then converted to its hexadecimal representation. For example, the hexadecimal representation of “a” is “61.” Each character is then converted to binary. At this binary level, each block is lined up against the key, and, character by character, the key and the block of the original message do a bitwise “exclusive-or” operation.
Here is a simple example.

Suppose our key is the simplistic “aaaaaaaaaa,” which is ten characters in length. When we send our message that begins with “abcdefghij,” the two sets of ten characters are lined up. First, “a” is compared with “a,” then “b” with “a,” then “c” with the next character of the key, which is again “a,” etc.

The binary representation of “a” is 01100001. This is compared with our own “a,” also 01100001. Wherever the two strings differ, a “1” appears; wherever they are the same, a “0” appears.

<table>
<thead>
<tr>
<th></th>
<th>01100001</th>
<th>01100001</th>
<th>01100001</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>a</td>
<td>a</td>
<td>(KEY)</td>
</tr>
<tr>
<td>a</td>
<td>01100001</td>
<td>b 01100010</td>
<td>c 01100011</td>
</tr>
</tbody>
</table>

Result: 00000000 = 00 00000011 = 03 00000010 = 02 (ENCODED)

So our first block of ten characters is converted from “abcdefghij” to the hexadecimal representation “0003020504070609080b”. Then, the first three characters (or six characters of hexadecimal) are shifted to the end, giving us “0504070609080b000302.” The process is then repeated with this new block of ten characters (20 in hexadecimal) five times, for a total of six rounds for each block.

This hexadecimal string of characters is sent to the recipient’s phone. The recipient's phone will then do the process in reverse, converting from hexadecimal to binary, running the six rounds backwards, and then showing the original message on the recipient’s screen. The entire process is outlined in an attached flowchart.

Most likely, the easiest way to implement this system is to make the key known to employees, who have to input the key as a password in order to send/receive text messages. However, for security purposes, this key should be changed frequently, probably at least daily.

As you can see, our system is able to take any text message and make it into a ciphertext that would mean nothing to anyone looking at it. By using hexadecimal representation, we are able to easily change the representation of each character. Then by the exclusive-or process, we
can encrypt the message so that it is even more unrecognizable. By shifting the characters and repeating this process for a total of 6 times for each block, we have created a powerful cryptosystem. This system will provide the security that your company is looking for through a process that is easy to implement into a phone.

Yours Sincerely,