“R22” – An Unbreakable, Un-hackable, Friend-to-Friend Encryption System

Robert Todd

Ernest C. Trefz School of Business

University of Bridgeport, Bridgeport, CT

Introduction:

This project begins a discussion of creating a secure “friend-to-friend” messaging scheme that involves only ordinary computers on the “surface web, does not depend on the difficulty of factoring, is impervious to any kind of deliberate or accidental backdoors, does not use any third party random number generators, and is unbreakable in any reasonable amount of time.

Outline of R22:

The proposed method uses offline, WiFi-unsupported computers to encrypt and decrypt messages. R22 uses a public 3-digit integer to create a random 20,000+ digit key, which is used to the alphabets for encryption. The plain-text is translated into a 29-character (A-Z, comma, space, period) which is then encrypted using at least 20 different alphabets, half of which are double-characters for a single plain-text character.

The encrypted plain-text is copied to a text document, transferred to an ordinary networked computer and transmitted openly, along with the public code.

Upon receipt, the text document is copied to the offline computer, the message copied from the text document, the key and the alphabets recreated fro the public code, and the message is decrypted.

Encryption Details:

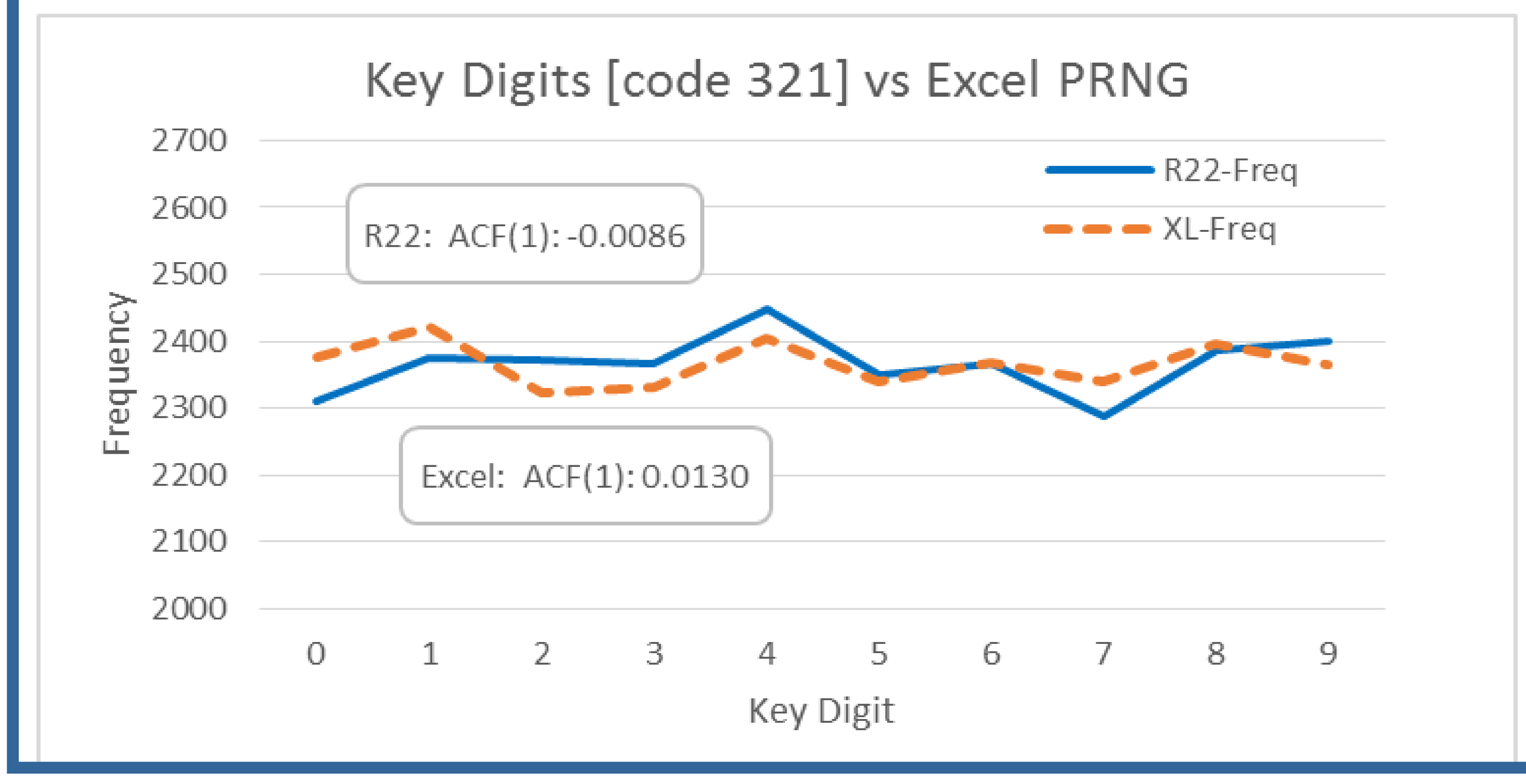
Create a secure key, K, using the public code, c:

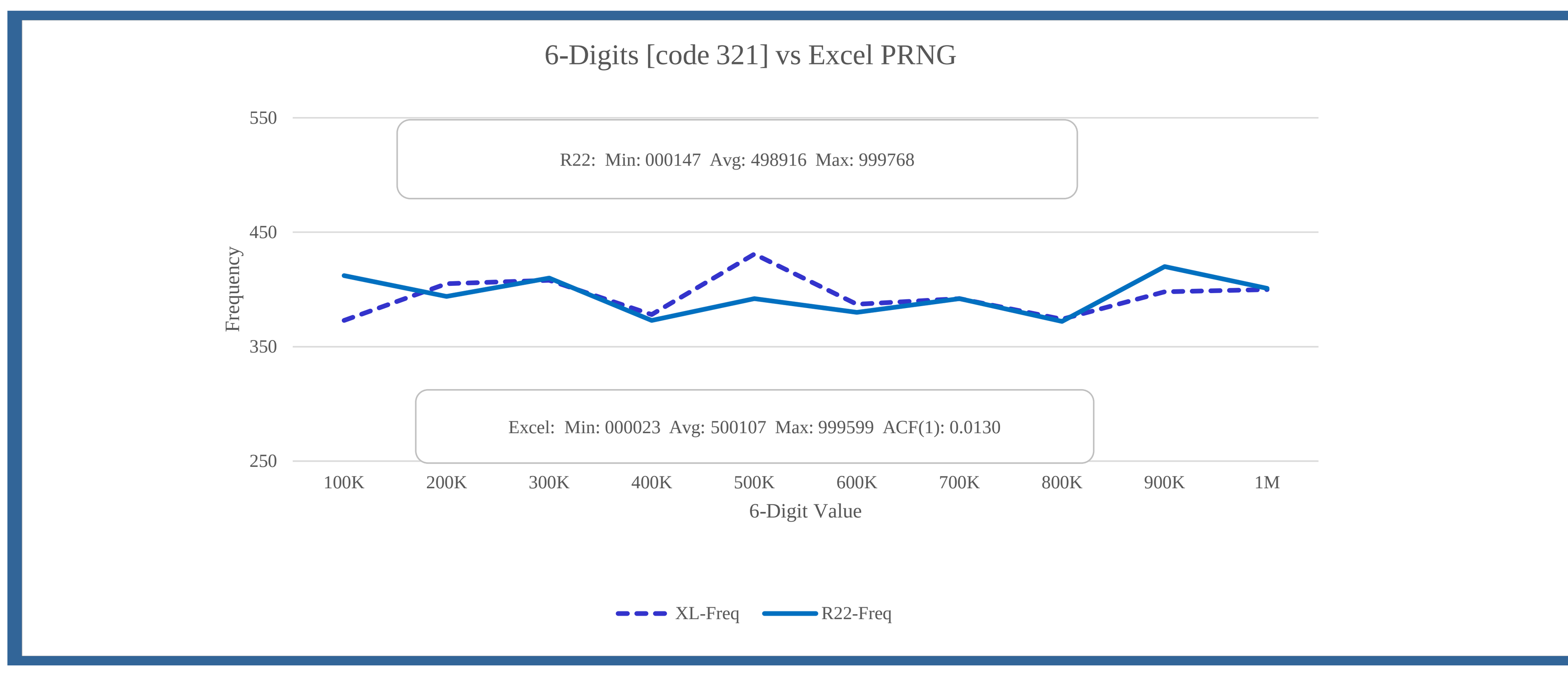
1. $K_0 = \sqrt{p}$ to many, many places, where p is some prime number
2. Choose 2 functions, $f_i(c)$ that will specify a starting point within $\sqrt{p}$, and a function $g(c)$ that will specify the length of K.
3. Create K_1 and K_2 as K_i = Mid $[\sqrt{p}, f_i(c), g(c)]$
4. Insert the digits of c into both K_1 and K_2 at different points, giving K_3 and K_4.
5. Create $K = K_3$ XOR K_4

Comments:

1. K must be, first, a PRNG. Then it must be a CSPRNG. At this point, all I’ve done is verify that the digits are uniformly distributed both individually (Fig.1) and when grouped into 6-digit sets (Fig. 2).

2. To be a CSPRNG, a PRNG must pass both the “next-bit” test and the “state compromise extension” test. One test for next-bit unpredictability is that there is no autocorrelation with a lag of 1. R22 passed that test for the 23,677 digits used (Fig. 1).





Encryption Details (cont.)

Translate plain-text to A-Z, space, period, comma.

Replace each non-alphabetic character with an appropriate equivalent, *e.g*, “@” → “ .AT. ” and each number with its spelled-out equivalent, as “4” → “ FOUR ”. (Note the spaces before and after each word.)

Use K to create the alphabets.

Assign the 29-character plain-text alphabet using Zipf’s Law and with the sum of frequencies at least 300. Use K to generate a random number between 100,000 and 999,999. This will be the Unicode value of each of 10x300 single characters and do it twice for each of the 10x300 double characters.

Use* c *and* K *to encrypt the translated plain-text.

Use K to determine the “single-double” sequence and, based on the result, use K to determine which alphabet and which of the multiple choices for each plain-text character.

Example of R22 Encryption:

Plain text: This is what R22 encryption looks like: ABC-123, Wow! What? This & that.

Encrypted text:

327

□□□□□□□□□□鬻□□□□□□纂□□□□□□□□□□□□□□增誏□□□□□□翙□□ĵ□□□
□□□□□□惆□□□
□□□□□□□□□□□□□蔓□□□□□□□□□□□务□挫□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□□□□□□□□椃□□□□□□飕□□□□□□□□□□夆□□□□□
□□□夆□□□□□筡□□□□□□□□□踌□□□□□

Here is the decrypted message.

THIS IS WHAT R TWO TWO ENCRYPTION LOOKS LIKE .COLON. ABC .DASH. ONE TWO THREE , WOW .XPT. WHAT .QM. THIS .AMP. THAT.

Physical Security:

The success of this method depends on keeping the friends’ choices secret. This is primarily accomplished by keeping the computer that encrypts and decrypts the messages remote, offline, and WiFi-disabled.

Weaknesses of R22:

Mostly the physical security required to keep the friends’ choices secret, but there are others – ask me!

Why R22 is nearly unbreakable:

No room on the poster – ask me!