Metalinguistically Hardened Caesar-shift Encryption

Will Styler
Chief Security Officer, the Society for the Prevention of Cruelty to Vowels
will@savethevowels.org

August 16, 2015

1 Introduction

'Caesar Shift' alphabetic substitution ciphers are among the best explored text encryption methods in the history of mankind. In such schemes, the characters in the alphabet used in a secret text are 'shifted' by a certain number of characters, turning an easily readable plaintext into a jumbled mess of letters, completely unreadable by such an adversary without access to a specialized shift factor.

Using a primitive version of such a cipher, a secret text ("This is top secret information") could be enciphered with a shift factor of four (turning each letter into a number, then adding four to each number, then turning the numbers back to letters), resulting in a completely unreadable cyphertext ("Xlmw mw xst wigvix mrjsvqexmsr").

Although such ciphers remained unbreakable throughout thousands of years of human history, alas, they suffer from several weaknesses.

2 Weaknesses of Conventional Caesar-Shift Approaches

Although they are suitably well-hardened to resist minute- or hour-scale attacks by 99% of the general population, conventional Caesar-shift approaches suffer from several weaknesses.

The most vexing problem is shortened key length. As it stands, for English language cyphertexts, the number of possible shift vector keys is limited to 25 (representing each of the possible shifts in a conventional cipher). This results in shift vector key lengths as small as 5 bits, compared to the modern standard key lengths of 128 bits, 256 bits, and even 4096 bits, which is a sizable weakness. This is addressed in MHCSE.

There is also a secondary attack which is possible in some circumstances when an adversary has detailed information about the linguistic nature of the plaintext, as well as incredibly detailed information about the nature of the lexicon in the language of transmission. In these rare cases, the attacker is able to model the possible plain texts corresponding to each key in such a way as to determine the most likely text based on complex linguistic knowledge.

It should be mentioned that, in order to conduct such an attack, the attacker needs to have sufficient linguistic knowledge to be able to parse and comprehend the plaintext. Depending on the language of transmission, this fact alone reduces the number of potential attackers considerably. Yet, many people have still raised some concerns about this possibility, resulting in 'simple alphabetic shift'

narrowly losing its bid to become the AES ("Advanced Encryption Standard") in 2001 (although the NSA was heavily involved in the decision to choose Rijndael over simple alphabetic shift. Infer from that what you will).

The addressing of these issues via a novel key use algorithm and, more importantly, metalinguistic hardening, results in a clearly superior algorithm, here called "Metalinguistically Hardened Caesar-shift Encryption" (MHCSE), which you should probably pay the author considerable money to implement.

3 MHCSE: An overview

Here, we will discuss Metalinguistically Hardened Caesar-shift Encryption here in a schematic and manual way. Implementation will be left to the individual users, as generally, implementation of algorithms introduces no security concerns at all.

3.1 Key Selection and Shift Factor Generation

First, a key must be generated. The key can be any arbitrary number of n length, generated by the Cryptographically Secure Psuedo-Random Number Generator (CSPRNG) of your choice, or by asking your friend in the next cubicle to call out numbers. We advocate using keys as large as possible, as the below shift factor generation procedure finally removes the onerous requirement that keys be shorter than or equal to the texts they encode.

We should also note here that although no key distribution mechanism is specified, the average carrier pigeon is capable of carrying digitized keys up to 1024000000000 bits, and unless your attacker is skilled with a shotgun, key interception risk is minimal, and such action generally alerts all parties involved, particularly the pigeon, that the key has been compromised.

But regardless, if we were to use the below 1024-bit key:

```
16152174667064029642647365822885998430666314431815268152405470907824573
65903662972483772980826569393306732864932303362619914669385966910731129
68626710792148904239628873374506302653492009810626437582587089465395941
37549600473991849827667633423824146549803003658606392990236819200423317
20320801887269656006171 67
```

We can calculate `[that large number] mod 25` to obtain the shift factor (here, 8). Note the use of modular arithmetic, a core component of many of the most secure algorithms available in the cryptographic world today, and its inclusion here clearly means that the author's hefty consulting fee is, in fact, a bargain.

3.2 Metalinguistic Hardening

For this example, we'll use the below top secret text:

We will meet behind the library at 2pm, where you will bring the secret documents and use the phrase "Winter is cold in Stockholm" to identify yourself as an agent.

As discussed previously, if we were to simply apply the shift factor to the plaintext at this point, we would suffer from the same woes as a conventional Caesar shift cipher: That is, that the message

can be modeled and retrieved using the assumption that the text is written in English. This is where metalinguistic hardening comes into play.

Rather than encrypting the message as it is, in a form subject to frequency and brute force attacks based on the known languge, we will transform the message into a metalanguage, which captures the essential meaning of the language, while removing the characteristics which allow analysis as an English plaintext.

To do this, we will rely on the below proprietary API call, which should not be reverse engineered, investigated, tested, and in fact, probably shouldn't be read at all lest you violate the terms of service implicitly agreed to in having read this paper. One needs simply to acquire a Google API key (for proprietary reasons) run the below:

```
https://www.googleapis.com/language/translate/v2?key=INSERT-YOUR-API-KEY
&q=[message to be encrypted]&source=en&target=es
```

This secret process, which again, should not be thought too much about, will generate a metalinguistic representation of the message based in a modern reimplementation of the Latin language used by the Caesars, for whom Caesar Shift ciphers were originally named. This representation will look like the below:

Nos encontraremos detrás de la biblioteca a las 14:00 , donde se traen los documentos secretos y utilizar la frase "El invierno es frío en Estocolmo" para identificarse como un agente

This representation resists attacks based on English frequency and language modeling, and will cause a brute force attack which attempts to find comprehensible English sentences to fail. As no humans are involved in the generation of the metalanguage, the process is also resistant to social engineering and other human vulnerabilities.

This already-incomprehensible metalinguistic representation is then alphabetically shifted according to the shift factor generated above, resulting in a completely impenetrable text:

```
Uvz lujvuayhyltvz klayáz kl sh ipispvaljh h shz 14:00 , kvukl zl ayhlu svz kvjbtluavz
zljylavz f bapspghy sh myhzl ``Ls pucplyuv lz myív lu Lzavjvstv'' whyh pkluapmpjhyzl
jvtv bu hnlual.
```

This message can then be sent as plaintext without fear of interception, decoding, or attack.

3.3 Decrypting MHCSE

At this point, the recipient must again derive the shift factor from the key (see 3.1), and reverse the shift process. This will yield the metalinguistic representation derived earlier.

This representation can then be transformed back towards the original text using a second proprietary API call (which again, for optimal security, should not be investigated, parsed, or read by human programmers):

```
https://www.googleapis.com/language/translate/v2?key=INSERT-YOUR-API-KEY
&q=[metalinguistic representation]&source=es&target=en
```

This results in a new version of the original text:

We will meet behind the library at 14:00, where the secret documents are brought and used the phrase "winter is cold in Stockholm" to identify as an agent

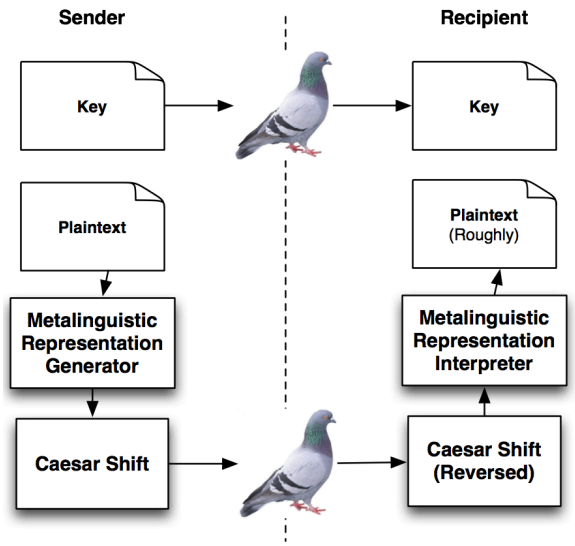

Figure 1: Schematic View of the MHCSE

Note that although the rough meaning is similar, this text differs slightly from the original plaintext. This is an important feature of the metalinguistic representation algorithm, referred to as *Perfect Backward Secrecy*. With PBS, even if an attacker is somehow able to intercept both the message and secret key, due to the intricacies of the metalinguistic representation algorithm, it cannot be proved that the message decoded is the same as that which was originally sent, and indeed, all prior messages are cast into doubt as well.

This provides a degree of plausible deniability for all parties, as these small variations allow you to answer truthfully that the decoded message is not the one which was sent, even when faced with a polygraph, legal challenge, or $5 wrench. Although the uncrackable nature of the cipher should leave this precaution unnecessary, it represents yet another advantage of this algorithm over conventional algorithms where the key and encoded message are deterministically related to the plaintext.

Thus, we see that with MHCSE, any text can be transmitted to the second party who is in possession of the proper key, securely and without the vulnerabilities associated with the traditional Caesar shift approach.

3.4 'Weaknesses' of MHCSE

While the author can assure you that there are no meaningful weaknesses with the algorithm itself, in order to make the other entrants in this contest feel better, we will highlight several features which, to the untrained eye, might appear to be faults.

First, for intensely proprietary reasons, all numbers requiring encryption must be entered as text ('four hundred fifty five' instead of '455'). This is a feature, as this increased effort would never be expected by an adversary, thus serving to make your plaintext less predictable, and thus, more difficult to discover. In addition, it is well-known fact that long-form numbers are more secure, hence their inclusion on checks.

For this reason, binary data (represented as "zero one zero zero one one zero...") is somewhat

differently space efficient with encrypted using MHCSE than inferior solutions (AES, ECC, Twofish, Enigma), but again, with the birth of incredibly large SD cards, message length is of little concern for modern carrier pigeons.

Also, it should be noted that as described, the algorithm will only work for encoding the English language. But, again, this is not a weakness, as the author, in exchange for consulting fees, will happily provide *deeply* proprietary replacement code functioning with romanized versions of many of the other major languages spoken around the world (Encipherment of the Spanish language requires specialized metalinguistic representations which differ from those discussed above, for reasons which are entirely proprietary and in no way germane to the discussion at hand).

Due to the specialized and highly technical nature of the generation of metalinguistic representations and the intricacies of Perfect Backward Secrecy, it is occasionally the case that the decoded plaintext is not interpretable as grammatical English. For this reason, we recommend that each message be sent several times, phrased slightly differently in each time. This is, again, a feature, as it helps to harden against accidental pigeon death, complicates quantum attacks considerably, and forces any attackers to struggle to decrypt many messages for each meaningful transmission.

Finally, it's worth pointing out that this is one of few crypto systems hardened against "Schneier's Law", a litmus test for spurious cryptography which states that *"Any person can invent a security system so clever that she or he can't think of how to break it."* Indeed, due to the deeply complicated and proprietary nature of the algorithm which you need not concern yourself with, the author *can* in fact think of how to break it. But he won't tell anybody. He swears. Thus, the algorithm is again shown to be robust and uncrackable.

Thus, we see that all 'weaknesses' of MHCSE are actually strengths. Which, if I were your CSO, I would find *very* reassuring. Thus, we can safely state that unless your company is using MHCSE for all data transmission, you are exposed to several hundred billion dollars in accidental data release lawsuits *right now*. Thus, really, you can't *not* afford to contact the author to discuss his consulting rates, which are very reasonable.

4 Conclusion

Given that the author used a cloud file synchronization service when writing this, it is indeed very likely that several state-level agencies have already abandoned their previous solutions and adopted MHCSE prior to submission, as it has certainly passed all relevant reviews for cryptographic soundness with flying colors.

Thus, we see that Metalinguistically Hardened Caesar Shift Encryption is, simply put, the very best approach to text encryption, and giving the author considerable sums of money is almost certainly the only way that you and your company will find a true sense of safety in the scary, scary world of information security.