Assignment 1 - Beating up on Old Ciphers

Cryptography

Robert Boerwinkle

Spring 2022-23

0 Seed

Because of the structure of the assignment, the seed used to generate the encodings needs to be reported: 2224252. This is not relevant to the project, simply a bookkeeping measure. This project goes through the cracking of several simple ciphers. It is written in Python using only the standard library.

1 Substitution

> wzm sqnqe
> qs q jmttu uxcai zcawsjqa rqs xaym ixdai ptdshnu qnxai wztxciz q rxxe, wzmtm yqjm ck q ndwwnm xne rxjqa, qae sqde wx zdj, 'ixxe equ, ixxe equ; uxc smmj jmttu maxciz, pcw d qj zcaitu qae wzdtswu; ex ktqu idbm jm sxjmwzdai wx mqw.' wzm zcawsjqa wxxh kdwu xa zmt, qae kcw zds zqae da zds kxyhmw qae iqbm zmt rzqw zm zqe. wzma zm rqawme wx ix zds rqu; pcw szm wxxh zxne xl zdj, qae sqde, 'ndswma, ju ltdmae, wx rzqw d qj ixdai wx wmnn uxc; d rdnn tmrqte uxc lxt uxct hdaeamss; ix uxct rqu, qa...

This text (`1_input.txt`) is encrypted using a substitution cipher. An arbitrary mapping of letters to letters has encoded this message (case insensitive). With a little cleansing and frequency analysis, we can determine the relative frequencies of each letter. Compare this to the relative frequencies of letters throughout the English language (from https://pi.math.cornell.edu/~mec/2003-2004/cryptography/subs/frequencies.html). The full report is available from the output of `1.py`.

total	986	818	751	682	619	580	560	462	455	386	361	246	226	218	...
letter	m	w	q	z	x	a	d	s	e	n	t	r	c	j	...
English	e	t	a	o	i	n	s	r	h	d	l	u	c	m	...

From this we can do a simple substitution:

> toe radah
> ar a melly yicnf ocntrman uar inge fisnf plsrbdy adinf tolicfo a uiih, toele game cv a dsttde idh uiman, anh rash ti osm, 'fiih hay, fiih hay; yic reem melly enicfo, pct s am ocnfly anh toslrty; hi vlay fske me rimetosnf ti eat.' toe ocntrman tiib vsty in oel, anh vct osr oanh sn osr vigbet anh fake oel uoat oe oah. toen oe uanteh ti fi osr uay; pct roe tiib oidh iw osm, anh rash, 'dsrten, my wlsenh, ti uoat s am fisnf ti tedd yic; s usdd leualh yic wil yicl bsnhnerr; fi yicl uay, an...

If you stand back, cock your head, and squint a little, this kind of looks like English. At this point, without a large dictionary of words and a lot of brute force, it is very difficult to get the rest of the way. Luckily, these types of ciphers are relatively easy to crack by hand with a significantly large text. The only one letter words in English are "I" and "A". The single letters that show up in the text are "s" and "a", implying every instance of "s" should be an "i". Similar clues can be gathered until the text can be fully decoded (`1_output.txt`).

> the salad
> as a merry young huntsman was once going briskly along through a wood, there came up a little old woman, and said to him, 'good day, good day; you seem merry enough, but i am hungry and thirsty; do pray give me something to eat.' the huntsman took pity on her, and put his hand in his pocket and gave her what he had. then he wanted to go his way; but she took hold of him, and said, 'listen, my friend, to what i am going to tell you; i will reward you for your kindness; go your way, an...

With a little searching, it appears to come from https://etc.usf.edu/lit2go/175/grimms-fairy-tales/3168/the-salad/. Wikipedia's article on the story says: "These tales are classified in the Aarne–Thompson–Uther Index as ATU 566, 'The Three Magic Objects and the Wonderful Fruits'."

2 Shift

> Zk jkp wyyalp wjupdejc xaywqoa ep ykiao bnki pda ikqpd kb w naolaypaz lanokj.
> – Xqzzdw

This text (`2_input.txt`) is far shorter, but uses a far simpler cipher: a shift cipher or Caesar cipher. In this cipher, every character is shifted a set number of places down the alphabet. For example, under a shift of 2, "my data" becomes "oa fcvc". This also uses frequency analysis. After the order of letters is determined, the 'distance' (https://en.wikipedia.org/wiki/Hamming_distance) from the English frequency is determined (`2.py`).

```
def getDistance(s0,s1):
        out = 0
        for i,letter in enumerate(s0):
                out += abs(i-s1.find(letter))
        return out
```

Each of the 25 possible shifts are tried, and the one with the shortest distance is chosen. This one happened to have a shift of 4 (`2_output.txt`). The distance turned out to be 90.

> do not accept anything because it comes from the mouth of a respected person.
> – buddha

3 Affine

This section has no text to decode. It simply asks for the substitution made by an affine cipher. The affine cipher is defined as: $(ax + b) mod 26$ where x is the index of the letter in the alphabet. This particular cipher has an a of 23 and b of 21. The substitution is shown here:

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
v	s	p	m	j	g	d	a	x	u	r	o	l	i	f	c	z	w	t	q	n	k	h	e	b	y

$(a^{-1}(y - b)) mod 26$ is used to decode this cipher (where y is the index of the encoded letter in the alphabet). Even to non-maliciously decode the cipher, a^{-1} needs to be found. This would require complex math, but there only 26 options. Brute forcing $(a \cdot a^{-1}) mod 26 = 1$ is a completely viable option. In this case, $a^{-1} = 17$. This can all be found in `3.py`, using `3_input.txt`.

4 Future Proofing

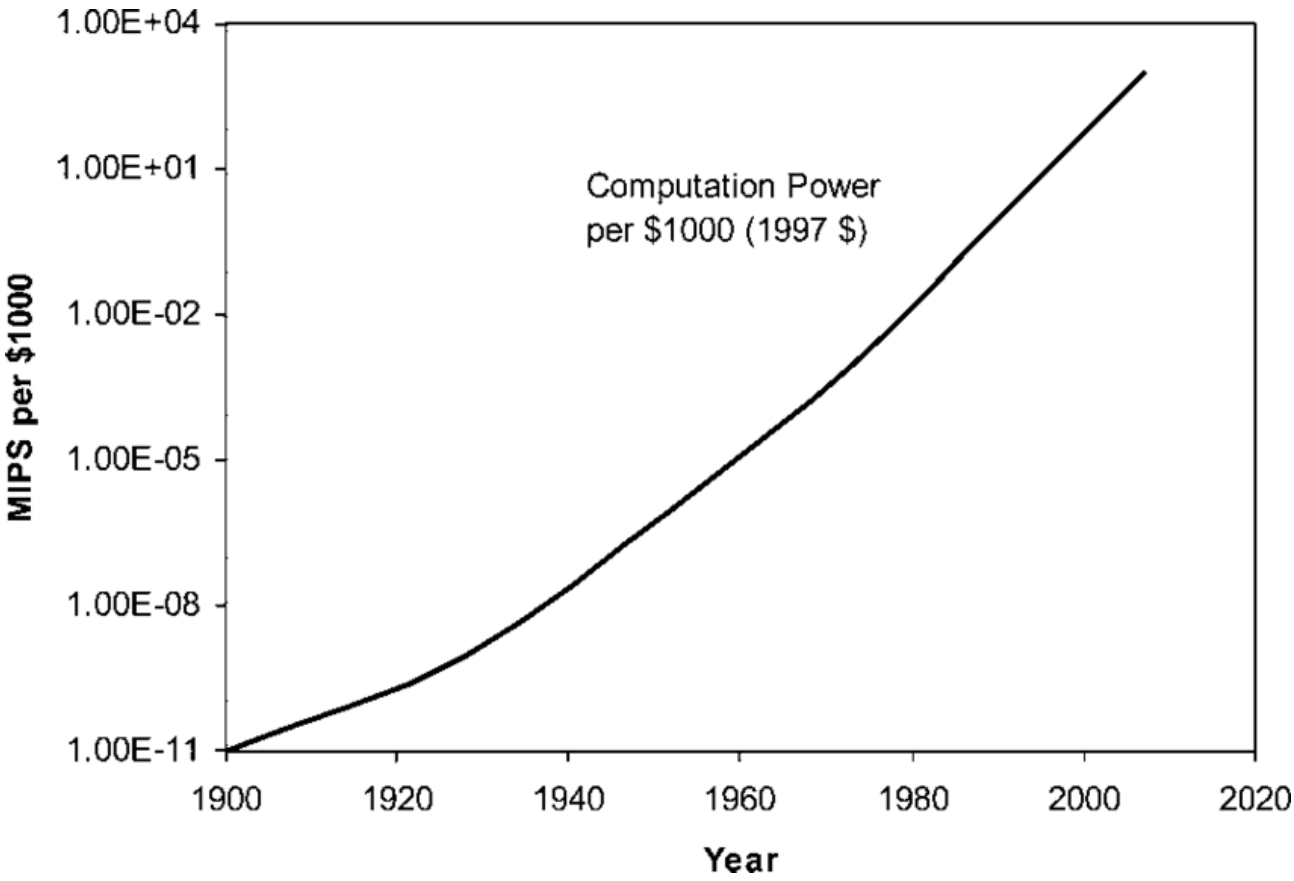

Figure 1: https://www.researchgate.net/figure/Change-in-the-PCR-for-computers-over-time-\E-exponent-MIPS-million-instructions-per_fig8_8914157

The question "If you can crack an encryption scheme today by spending $1,000,000, how long until it can be cracked for $100?" was posed in the assignment, with the caveat of only including brute force attacks. This graph has interesting units: million instructions per second per $1000. This is perfect assuming a similar time frame is required to crack the encryption. It seems to suggest that the answer is approximately 30 years (assuming no inflation). People like to talk about quantum computers and how they will break encryption as we know it. This may well be true, but speculation about such possibilities is far beyond the scope of this paper. In any case, the change will be gradual as quantum computers become cheaper and more accessible. Hopefully infrastructure will be able to adapt in time.