Fall 2019
Chris Christensen
MAT/CSC 483
Section 5

Keyword Ciphers

Look at the following random MASC key for a minute.

```
abcdefghijklmnopqrstuvwxyz
KPFHIGLDEXCVTOUBJQZMRNAYSW
```

Now, cover up the key and write it from memory.

Unless you have a remarkable memory, it is unlikely that you would be able to remember and recreate the plaintext-ciphertext correspondence for this; it is likely you and the receiver would have to write down the key. Having to write down the key jeopardizes key security.

Affine ciphers (including Caesar ciphers and multiplicative ciphers) have short and, therefore, memorable keys, but the number of keys is small and, although they might not be easy to spot (in the cases of multiplicative and affine ciphers), patterns are introduced into single letter frequencies. Another scheme that uses a memorable key for a simple substitution cipher is called the **keyword cipher**. Its key consists of an easily memorized word or phrase and a letter. Here is how one common version of the keyword cipher works.

Cryptography

The key has two parts – a word or phrase and a letter of the alphabet.

1. Select a keyword or phrase.

```
Northern Kentucky University
```

   and a keyletter

```
j
```

2. Reading from left to right, write the word or phrase without duplicating letters.

```
NORTHEKUCYIVS
```

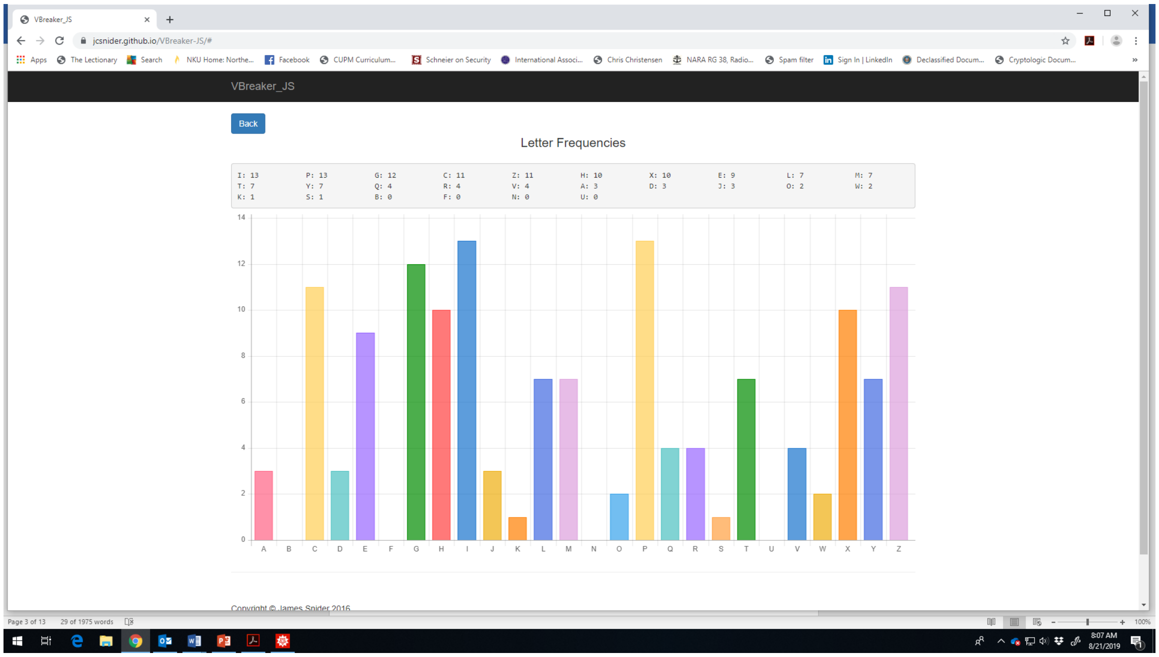
3. Underneath the plaintext alphabet, beginning with the keyletter, write, letter for letter, the keyphrase with the duplicate letters removed. After the last keyphrase letter, write the so far unused letters of the alphabet in their usual order – cycling back to the beginning.

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
G	J	L	M	P	Q	W	X	Z	N	O	R	T	H	E	K	U	C	Y	I	V	S	A	B	D	F

Here is a message encrypted using a simple substitution cipher and the key given above.

```
ivhhd agyix paecm vypmg ijrpi lxrpd kgcoi empyl
czjpi xpicg qqzle hlpci gzhqz yxrzh oyaxz lxcgm
zgipm qceti xpwpc tghxz wxlet tghmz hjpcr zhevi
ieixp sgcze vygct dlett ghmy
```

Frequency analysis shows the following single-letter frequencies.



```
I: 13    P: 13    G: 12    C: 11    Z: 11    H: 10    X: 10    E: 9     L: 7     M: 7
T: 7     Y: 7     Q: 4     R: 4     V: 4     A: 3     D: 3     J: 3     O: 2     W: 2
K: 1     S: 1     B: 0     F: 0     N: 0     U: 0
```

The frequencies show the peaks and valleys that should be expected with a MASC, but the distribution of those peaks and valleys do not correspond to plaintext or to a Caesar shift. (They also do not correspond to a multiplicative cipher, but that is harder to determine.)

Keyword ciphers have a large keyspace, but some keys leave long strings of letters in alphabetical order in the ciphertext alphabet and some keys leave many letters of plaintext fixed.

For example, if our keyword were `computer science` and our keyletter were `g`, our key would be:

```
abcdefghijklmnopqrstuvwxyz
QVWXYZCOMPUTERSINABDFGHJKL
```

Notice that once the keyword ends, there are strings of letters that are nearly in alphabetical order – especially near the end of the alphabet.

Here is even a worse case.

If our keyword were `bad` and our keyletter were `a`, we would have the key:

```
abcdefghijklmnopqrstuvwxyz
BADCEFGHIJKLMNOPQRSTUVWXYZ
```

Many letters are left fixed by this key.

Shifting the keyletter helps

```
abcdefghijklmnopqrstuvwxyz
WXYZBADCEFGHIJKLMNOPQRSTUV
```

but there is still an extremely long string of ciphertext letters in alphabetical order. This key is very similar to a Caesar cipher with additive key 4. We would prefer to have a keyword that resulted in a more random arrangement of ciphertext letters.

Using the Pattern in the Key for Cryptanalysis

So, we can create a memorable key for a simple substitution cipher, but, there is a trade off. We have created a memorable key at the expense of having created a pattern – we have placed a pattern in the key.

We noted in the keys above that keys may have long strings of ciphertext letters in alphabetical order. This pattern can be exploited by the cryptanalyst.

Before doing a complete cryptanalysis of a ciphertext message, we will consider how using the pattern in the key can help the cryptanalyst.

Assume that we have a ciphertext message that we suspect was encrypted with a keyword cipher and that we been able to partially solve it. Assume that we have recovered this much of the key.

```
    abcdefghijklmnopqrstuvwxyz
      Z EN   Y B FG  JLM   R V
```

Recall how the key is created for a keyword cipher. `Z` is unlikely to be in the keyword; so, we might suspect that the keyword begins immediately after `Z`. `FG` might indicate that the keyword has ended and we are seeing the "unused letters of the alphabet in their usual order." Looking back a few letters, it seems reasonable to assume that `Y` ends the keyword. If we are correct in assuming that `Y` ends the keyword, then plaintext `k` must correspond to ciphertext `A`. Put that in place.

```
    abcdefghijklmnopqrstuvwxyz
      Z EN   YAB FG  JLM   R V
```

`E` appears in the keyword; so, the letter between `B` and `F` is either `C` or `D`, which means that either `C` or `D` is in the keyword. Plaintext `p` must correspond to ciphertext `H` and `q` to `I`. `u` corresponds to `O`, `v` corresponds to `P`, and `w` corresponds to `Q`. Two of `STU` must be in the keyword. `a` must correspond to `W`, and `b` to `X`.

Here is what we suspect.

```
abcdefghijklmnopqrstuvwxyz
WXZ EN   YAB FGHIJLMOPQR V
```

and K is in the keyword, either C or D is in the keyword, and two of STU are in the keyword.

```
_ E N _ _ _ Y
```

Doesn't _ E N suggest KEN? If so, the keyword might be KENTUC(K)Y.

The pattern in the keyword helped us complete the key.

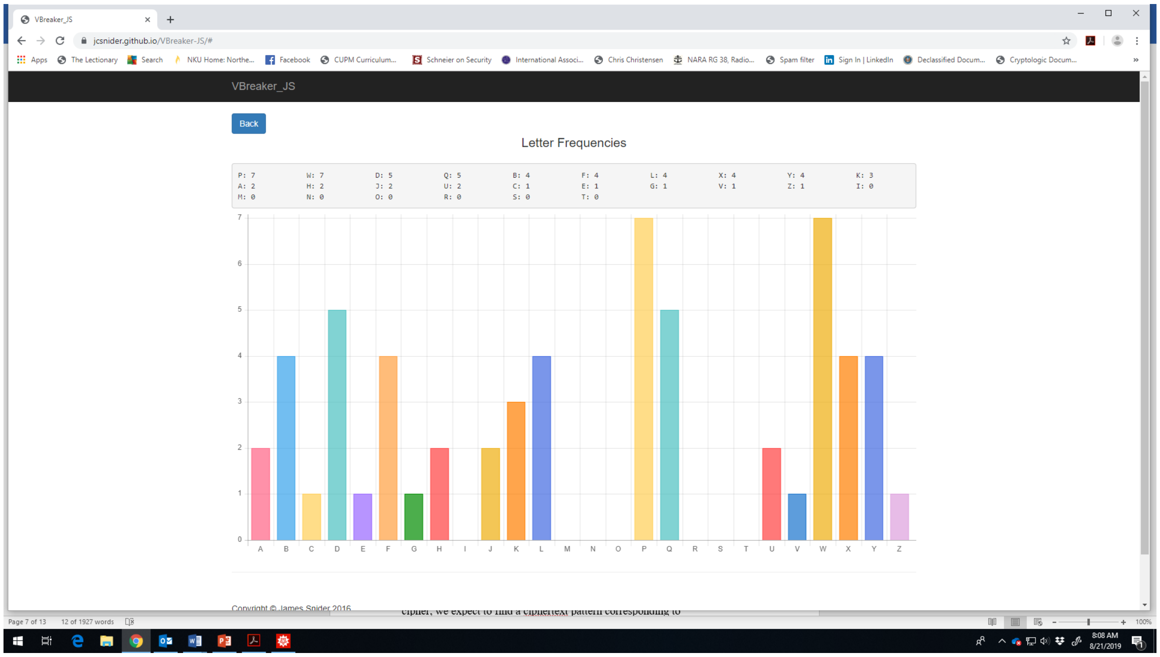
```
abcdefghijklmnopqrstuvwxyz
WXZKENTUCYABDFGHIJLMOPQRSV
```

Cryptanalysis Using Known Plaintext

Here is a ciphertext message:

```
PYWPX   FBWAK   PVGUF   DPXFB   PHQDE   WXXWK   DPYDL
HQZLY   QFLPY   JQDQU   LABWK   JBWWC
```

Cryptanalysis of any ciphertext message begins with frequency analysis.



```
P: 7     W: 7     D: 5     Q: 5     B: 4     F: 4     L: 4     X: 4     Y: 4     K: 3
A: 2     H: 2     J: 2     U: 2     C: 1     E: 1     G: 1     V: 1     Z: 1     I: 0
M: 0     N: 0     O: 0     R: 0     S: 0     T: 0
```

Copyright © James Snider 2016

The frequencies have peaks and valleys; so, we suspect that a MASC was used. The arrangement of the peaks and valleys suggests that a Caesar cipher was not used.

We have a crib. We suspect that the plaintext message contains the name `Thomas Jefferson`. Because this appears to be a MASC, we expect to find a ciphertext pattern corresponding to

```
_ _ 1 _ _ 2 _ 3 4 4 3 _ 2 1 _
t h o m a s j e f f e r s o n
```

We search the ciphertext for such a pattern, and we find one.

PYWPX   FBWAK   PVGUF   DPXFB   **PHQDE**   **WXXWK**   **DP**YDL
HQZLY   QFLPY   JQDQU   LABWK   JBWWC

From the crib we obtain a substantial amount of the plaintext and a substantial amount of the key.

```
oneof   the r   o   t   so th   omasj   effer   sons
PYWPX   FBWAK   PVGUF   DPXFB   PHQDE   WXXWK   DPYDL

ma  n   at on    asa      her    hee
HQZLY   QFLPY   JQDQU   LABWK   JBWWC
```

```
abcdefghijklmnopqrstuvwxyz
Q   WX B E  HYP  KDF
```

Look at the partial key. It appears to have been created by a keyword.

X_B and the placement of Y suggests that Y is in the keyword, that XZB appears as a ciphertext string, and that the keyword begins with B.

DF suggests that the keyword ends with K. The keyword appears to be

B _ E _ _ H Y P _ _ K

If we are correct about our assumptions, A and C must be in the keyword, and two of RSTUV are in the keyword.

From

```
t_on
FLPY
```

it is reasonable to expect that plaintext i corresponds to ciphertext L.

Then the key becomes

```
abcdefghijklmnopqrstuvwxyz
Q   WX BLE  HYP  KDF
```

P _ _ K knowing that a must be in the keyword suggests PARK.

```
abcdefghijklmnopqrstuvwxyz
Q   WX BLE  HYPARKDF
```

The keyword is likely to be BLETCH(LE)YPARK. That is the correct key.

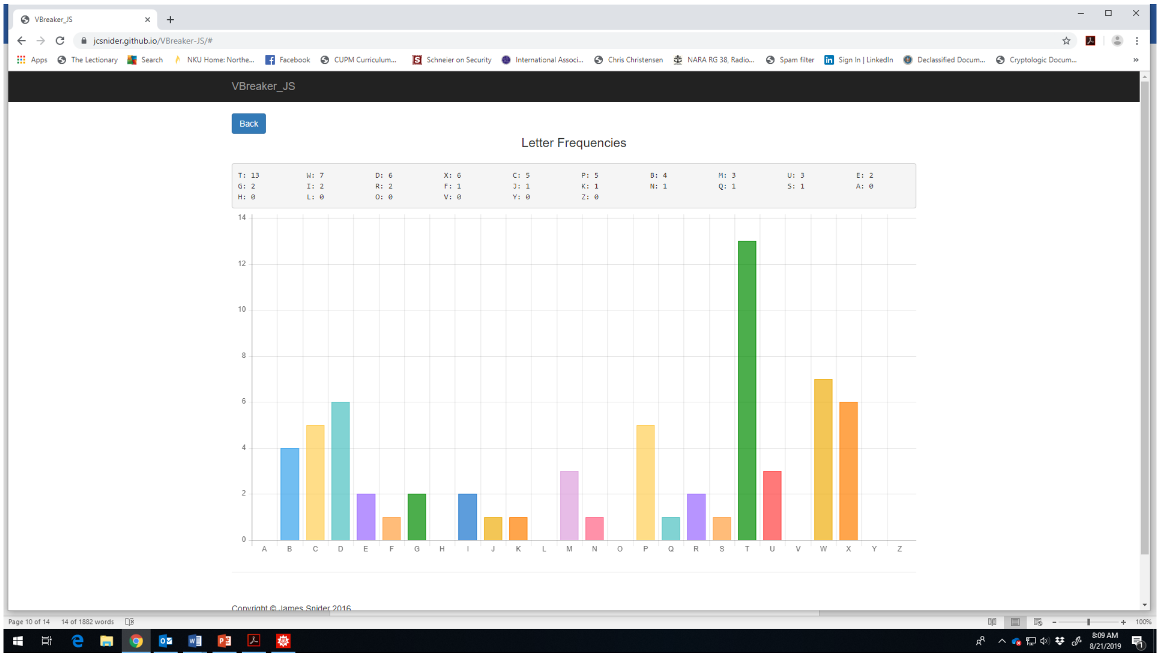
Cryptanalysis Using a Ciphertext Attack

Here is a ciphertext message:

```
DWTRF   TCPBX   CGITT   KPNME   TGUPC   WXUDR   XMWTB
JWTBT   DWTTI   DXBTP   EMWPQ   TDXCC   WXUDT   S
```

Here is its frequency analysis.

Single-letter frequencies:


VBreaker_JS
jcsnider.github.io/VBreaker-JS/#
Back
Letter Frequencies
14
12
10
8
6
4
2
0
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
Copyright © James Snider 2016

```
T: 13    W: 7    D: 6    X: 6    C: 5    P: 5    B: 4    M: 3    U: 3    E: 2
G: 2     I: 2    R: 2    F: 1    J: 1    K: 1    N: 1    Q: 1    S: 1    A: 0
H: 0     L: 0    O: 0    V: 0    Y: 0    Z: 0
```

Again, the peaks and valleys suggest a MASC but not a Caesar cipher.

Let’s also gather information about the frequencies of strings of 2 or 3 letters.

The most frequent bigraphs (2-letter strings) are:

`WT` 4 times

`UD, DX, DW, MW, TB, TT, TD, XC, XU, BT, CW` two times each.

The most frequent trigraphs (3-ltter strings) are:

`CWX, WXU, DWT, WTB, XUD` twice each.

As we collect more information, we will *pay attention to the recovered key*. If it appears to be a keyword cipher (and it will), we will use what we know about the construction of those keys to help with the cryptanalysis.

Because `T` is the most frequent ciphertext letter, `DWT` is one of the most frequent trigraphs, `D` is a frequent ciphertext letter, and `DW` is a frequent bigraph; ciphertext `DWT` likely corresponds to plaintext `the`.

```
the    e        ee          e      h  t     he
DWTRF  TCPBX  CGITT  KPNME  TGUPC  WXUDR  XMWTB

 he e  thee   t  e     h    et     h  te
JWTBT  DWTTI  DXBTP  EMWPQ  TDXCC  WXUDT  S
```

```
    abcdefghijklmnopqrstuvwxyz
        T  W           D
```

Notice

```
efgh
T  W
```

This suggests that `U` and `V` fit between `T` and `W` and that we are cryptanalyzing a keyword cipher. If we are correct, `f` corresponds to `U` and `g` corresponds to `V` and that this is part of the long string of letters at the end of the alphabet that often appear in alphabetical order in a keyword cipher.

```
    abcdefghijklmnopqrstuvwxyz
        TUVW           D
```

We will assume that `XYZ` do not appear in the keyword so the key looks like

```
    abcdefghijklmnopqrstuvwxyz
        TUVWXYZ        D
```

After substituting in the ciphertext, the partially decrypted message looks like:

```
    the    e   i     ee         e      hift   i he
    DWTRF  TCPBX  CGITT  KPNME  TGUPC  WXUDR  XMWTB

     he e  thee   ti e     h    eti    hifte
    JWTBT  DWTTI  DXBTP  EMWPQ  TDXCC  WXUDT  S
```

Notice

```
    thee   ti e
    DWTTI  DXBTP
```

`e_ti_e` might be `entire`.

If so, the partial key is:

```
        abcdefghijklmnopqrstuvwxyz
            TUVWXYZ  I   B D
```

and the partially decrypted message is:

```
    the    e  ri    nee         e      hift   i her
    DWTRF  TCPBX  CGITT  KPNME  TGUPC  WXUDR  XMWTB

     here  theen  tire     h    eti    hifte
    JWTBT  DWTTI  DXBTP  EMWPQ  TDXCC  WXUDT  S
```

From the partial key, it appears that s corresponds to C and that the keyword is likely to be between Z and B: _ _ I _ _ _ .

Substituting s for C results in:

```
the    es ri  s nee         e   s  hift   i her
DWTRF  TCPBX  CGITT  KPNME  TGUPC  WXUDR  XMWTB

 here  theen  tire     h    etiss  hifte
JWTBT  DWTTI  DXBTP  EMWPQ  TDXCC  WXUDT  S
```

The last letter in the ciphertext message S seems to correspond to d,

s nee
CGITT suggests that o corresponds to G, and

hift   i her
WXUDR   XMWTB suggests that c corresponds to R and p corresponds to M.

If we are correct, the partial key now looks like

```
abcdefghijklmnopqrstuvwxyz
  RSTUVWXYZ  IGM BCD
```

and the keyword is _ _ I G M _.

ENIGMA seems to leap out, and it is the keyword.

```
abcdefghijklmnopqrstuvwxyz
PQRSTUVWXYZENIGMABCDFHJKLO
```

The plaintext is:

> the cuesar is one example of a shift cipher where the entire alphabet is shifted

In this example we combined information gained from frequency analysis, from the partial decrypts, and from the partial keys to cryptanalyze the message.

As you cryptanalyze a ciphertext message, try to recover as much of the key as possible. Looks for patterns in the key as well as patterns in the plaintext.

Brute Force

If the key consists of one dictionary word, it is possible to find it by brute force using a dictionary attack – use a computer to try decrypting the message using each word in a dictionary as the possible key word and each letter of the alphabet as the keyletter. It is not elegant, but it works. This argues for using less predictable key phrases rather than keywords.