CS 409 Chalk and Talk

Authors:

Mradul Kaushal	23b1255
Avaneesh Vinayak Khadye	23b2124

September 30, 2024

Abstract

This report delves into our Chalk and Talk Topic, **On Project Venona**. Following a comprehensive 30-minute presentation, We now articulate our findings in written form. Our research centers on Multi-Time Pads, a cryptographic method of significant historical interest. We introduce an original attack strategy that leverages **frequency analysis**, the index of coincidence, and the cosine angle test to successfully recover complete messages encoded in Multi-Time Pads. Furthermore, we contextualize our analysis within the broader framework of the Venona project, highlighting its implications for cryptography and intelligence operations during the Cold War era.

Contents

Insecurity of Multi-time Pads

Perfectly Secure?

- Ciphertext reveals **no information** about plaintext.
- Acheived only if, $|K| >= |M|$
- Vernam Cipher is a **perfectly secure** cipher with $|K| = |M|$

$$Enc(k, m) = k \oplus m = c$$
$$Dec(k, m) = c \oplus k = (m \oplus k) \oplus k = m$$

Multi-Time Pad

The Vernam Cipher is also called **One Time Pad(OTP)** as the key can be used only once. As long as the communicating parties use the shared secret key **only once**, their communication is perfectly secure. Reusing the key makes the cipher completely insecure.

$$Enc(k, m_1) = k \oplus m_1 = c_1$$

$$Enc(k, m_2) = k \oplus m_2 = c_2$$
$$c_1 \oplus c_2 = (k \oplus m_1) \oplus (k \oplus m_2) = m_1 \oplus m_2$$

If 2 messages, m_1 and m_2 were encrypted by same key k, then the adversary can just XOR the ciphertexts, revealing information on m_1 and m_2, breaking perfect secrecy!!.

Insecure?

As an adversary, eavesdropping on a two time pad encryption allows us to obtain the XOR of two messages. How severe is this vulnerability, and to what extent can we reconstruct the original messages? Let's explore this with an illustration.

Figure 1.1: Images m_1 and m_2

Suppose that we use the same binary key for encryption of both the binary images. We now look from perspective of the eavesdropper and understand how much information about the images is leaked.

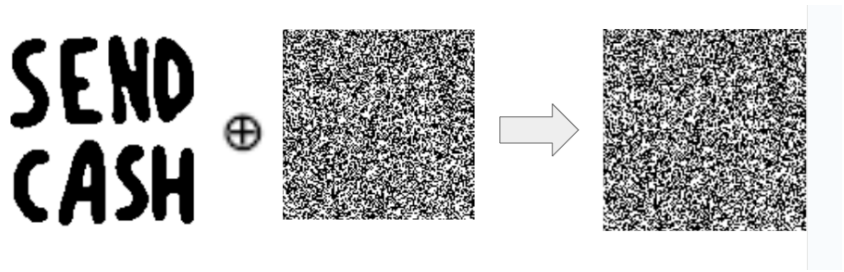

Figure 1.2: Encryption of m_1

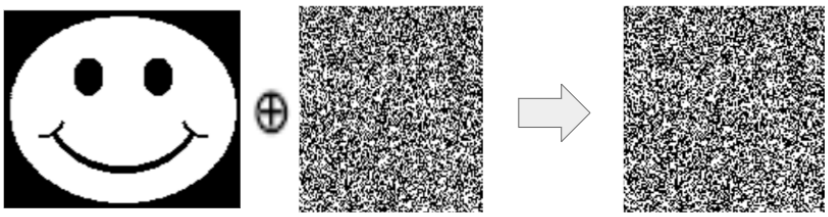

Figure 1.3: Encryption of m_2

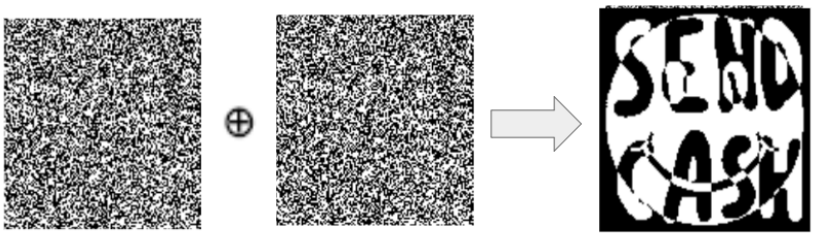

Figure 1.4: XORing the ciphertexts

We can conclude that Reusing the key is a very **BAD** idea.

MTP Breaks in Present Day

- The study on Multi-Time Pads is essential not only for cryptography but more so for fields like Natural language Processing and Automatic Speech Recognition.
- Modern algorithms use various advanced tools like HMMs(Hidden Markov Models), Smoothed n-gram language algorithm's and dynamic programming.
- Modern machines can **completely break 3-time pad**.

Code to break the Multi-Time Pad

In the following section, we shall demonstrate a Python program which uses frequency analysis to decrypt the Multi Time Pad.

Assumptions

- We assume that some message has been encrypted by a randomly generated key of some length. The encryption is carried out by computing the bitwise-XOR of each character of the message with each character of the key. The same key is repeated until its length matches that of the message
- We assume that the message length is much longer ($\approx$ 100 times) that of the key.

Encryption Scheme

Here is our assumed encryption scheme of security parameter 'length':

```
import random
def gen(length):
  key=''
  for i in range(length):
    key+=chr(random.randint(0,256))
  return key

def enc(m, length):
  key=gen(length)
  c=''
  for i in range(len(m)):
    c+=chr(ord(m[i])^ord(key[i%len(key)]))
  return c, key

def dec(c, key):
  m=''
  for i in range(len(c)):
    m+=chr(ord(c[i])^ord(key[i%len(key)]))
  return m
```

Using this encryption scheme, we could encrypt any long enough string "text" after cleaning up any whitespaces or punctuations present and converting all alphabets to lower.

```
text = ''.join(filter(str.isalpha, text)).upper()
c,key=enc(text,10)
```

Here, we have used a security parameter of length 10.

Steps to break the Cipher

- Find length of the key using Index of coincidence
- Use Frequency Analysis

1. Index of Coincidence (IoC)

The index of coincidence (IoC) is a statistic that measures how close a frequency distribution is to a uniform distribution. The IC is calculated by comparing the actual number of coincidences in a piece of text to the number of coincidences that would be expected by chance. The Index of Coincidence (IoC) is given by the formula:

$$IC = \frac{z \times \sum_{i=1}^{z} f_i(f_i - 1)}{N(N-1)}$$

Where:

- f_i is the frequency of each letter i in the text.
- N is the total number of letters in the text.
- z is the number of unique letters

In English, there are only 26 alphabets so z would be 26. Hence, using the formula, if we calculate the IoC for any general English text, we notice that it is around 1.6 .

Moreover, note that a ciphertext that has been encrypted using a singular character would have the same frequency distribution as English, albeit with different characters. Thus such a ciphertext would also have a IoC of around 1.6 .

So, using this information, what we do is take slices of our ciphertext, that is, we take characters from our encrypted string at regular intervals of some period k and compute the IoC(assuming z=26) of the slice. We do this by varying k from 1 to 50 (assuming key length is between 1 and 50).

By doing this, what we will notice is that when k is equal to *length*, the IoC of the slice will spike to around 1.6 . This is because when k is equal to key length, the resulting slice is nothing but a singular character encryption.

The code for calculating index of coincidence, assuming English:

```
from collections import Counter

def index_of_coincidence(text):
    frequencies = Counter(text)
    N = len(text)
    numerator = 0
    for f in frequencies.values():
     numerator+=f*(f-1)
    denominator = N * (N - 1)
    if denominator == 0:
        return 0
    IC = numerator / denominator
    return IC*26
```

Now we use this function and our aforementioned strategy:

```
import copy
found = False
graph=[0]*51
for k in range(1, 51):
 all_slices = ['']*k
 for i in range(len(c)):
  all_slices[i%k] += c[i]
 sum = 0
 for i in range(k):
    sum += index_of_coincidence(all_slices[i])
 ioc = sum / k
 graph[k]=ioc
 if ioc > 1.6 and found==False:
  period=k
  slices=copy.deepcopy(all_slices)
  found=True

import matplotlib.pyplot as plt
plt.bar(range(len(graph)), graph)
plt.show()
print(period)
```

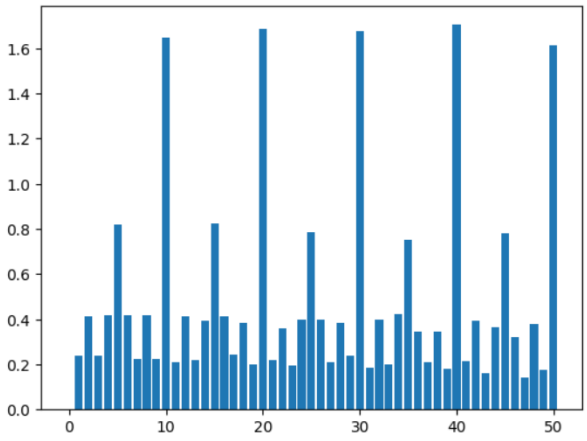

Figure 2.1: The graph we obtain after executing the above code

We notice that the first time when our IoC spikes to 1.6 is when k is equal to 10. This means the key length must be 10. We also see that the IoC spikes when k is a multiple 10. This happens since slices of the ciphertext taken at intervals of multiples of key length are also single character encrypted ciphertexts.

2. Frequency analysis

After we have obtained our slices, our next step is to do their frequency analysis by comparing them to English. For that we first need the frequency distribution of a general English text. In order to do so, we take the first chapter of "Tale of two cities" as $reference$. However one might take any sufficiently large 'correct' English text as $reference$.

```
monofrequencies = [0]*26
ALPHABET=[x for x in "ABCDEFGHIJKLMNOPQRSTUVWXYZ"]
for char in reference:
 x = ALPHABET.index(char)
 monofrequencies[x] += 1
for i in range(26):
 monofrequencies[i] = monofrequencies[i] / len(reference)
plt.bar([x for x in ALPHABET], monofrequencies)
```

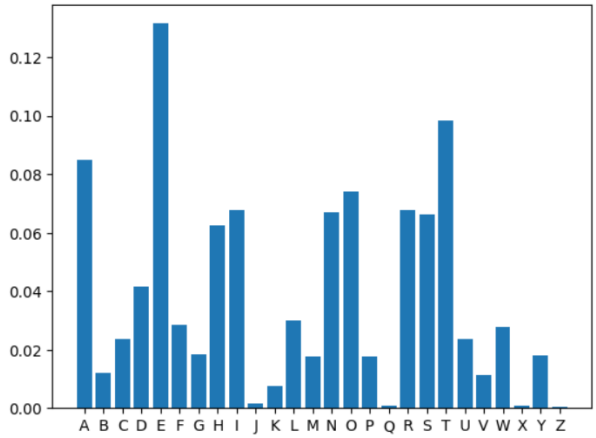

Figure 2.2: Monoalphabetic distribution of general English

The second tool we use for our analysis is the cosine function. It is used to see how close two vectors are. We will be using this to see how close our guess is to the actual frequency distribution of English. A cosine value of greater than 0.9 indicates that the vectors are very close.

```
from math import sqrt
def cosangle(x,y):
 numerator = 0
 lengthx2 = 0
 lengthy2 = 0
 for i in range(len(x)):
  numerator += x[i]*y[i]
  lengthx2 += x[i]*x[i]
  lengthy2 += y[i]*y[i]

 return numerator / sqrt(lengthx2*lengthy2)
```

Now that we have all the required tools, we shall complete our analysis. We know from our previous analysis that the length of key is some l. Hence, we have l different slices of our ciphertext corresponding to each character of the key. We shall analyse each slice independently to obtain the entire key.

While analysing the slice, we make our first guess, which is that the character that is occurring the most in our slice corresponds to 'E', the character that occurs the most in English. Taking this guess, we compute the XOR of the character with 'E' to obtain the probable character of our key. Using this 'probable character', we compute what the other characters in the slice might correspond to. Thus, we obtain our guess array, which contains the frequency distribution of the message, if the given slice had been encrypted by the 'probable character'.

We compare our guess array with the frequency distribution of English using the cosine function. If close enough, we assume our guess was right. If not, we take the guess that the character that is occurring the most in our slice corresponds to 'A', the the second most occurring character in English and repeat our steps.
As we can see, we can take 26 most guesses for each slice. In this way, we shall obtain every character of our key.

```
frequencies = []
for i in range(period):
 frequencies.append([0]*256)
 for j in range(len(slices[i])):
  frequencies[i][ord(slices[i][j])] += 1
 for j in range(256):
  frequencies[i][j] = frequencies[i][j] / len(slices[i])
 
 


key = ['A']*period
for i in range(period):
 monofreq=copy.deepcopy(monofrequencies)
 for j in range(26):
  e=monofreq.index(max(monofreq))
  g=frequencies[i].index(max(frequencies[i]))
  guess=(65+e)^g
  guess_array=[ord(x)^guess for x in ALPHABET]
  testtable=[frequencies[i][y] for y in guess_array]
  if(cosangle(monofrequencies,testtable)>0.9):
    key[i]=chr(guess)
    break
  monofreq[e]=0
```

We finally use the decryption function to break the cipher and obtain the message!

```
print(dec(c,key))
```

The Unbreakable Soviet код

Soviet Communication

Soviet agencies in the United States during World War II used three channels to communicate with Moscow.

- **Diplomatic Pouch**:- Most material went through this medium.But in wartime the pouch, though secure, was quite slow, often taking many weeks and sometimes months to reach its destination.
- **International Telegraphic Cables**:- Much of the time-sensitive info was channeled through this. Despite US and other nations routinely keeping copies of cables,Russians still used it as they believed their cipher to be unbreakable.
- **Short wave Radio**:-This was rarely ever used. The technology of the early 1940's did not allow for reliable short wave radio links between Moscow and North America unless one was using a large and powerful transmitting station, which the Soviets could not build within the confines of their diplomatic compounds. The Soviets equipped their American diplomatic posts with short-wave stations as backup for for emergencies such as breakdowns in commercial telegraphic traffic.

The Soviet Cipher

World War 2 lead to increased espionage which in turn lead to countries developing more secure ciphers.But increased espionage also meant that countries now invested more in intelligence projects to break the enemy's cipher. The nazis used ENIGMA, a complicated cipher that was thought to be unbreakable. The breaking of ENIGMA, through an collective effort by allied powers was a turning point in the War. The case with Soviets though was different. The Soviets used the One Time Pad, which was proven to be unbreakable. We now look at the pride of soviet espionage, their unbreakable cipher system.

- **KGB**:- Committee of Main Security, Soviet Union. One of its roles was to facilitate communication between Soviet spies in the U.S. and their headquarters in Moscow.
- **Int. Officer**:- A man of high command. He writes the message in concise Russian. He also converts any proper names or places mentioned to their code names.
- **Cipher Clerk**:- Uses a code book,to convert message to blocks of 4 digit numerals. He then performs the encryption algorithm and transmits ciphertext over the cable.

Let's understand the Encryption through an illustration,

- The **KGB** want to send the message, "**Ullmann delivered report about rockets.**" to Moscow Headquarters. They contact the **Int.Officer**.
- The **Int.Officer** writes down the message in concise Russian and substitutes Ullmann's name with his cover name(**Pilot**). He then hands the cipher clerk a message saying, "**Pilot delivered report about rockets.**"
- The cipher clerk consults a code book(a dictionary with precise instructions on how to convert words/punctuation/characters to 4 digit numerals) and converts the message to 4 digit numerals.

Pilot	delivered	report	about	rockets
7934	2157	1139	3872	2166

- The clerk then converts these four-digit groups into five-digit groups by shifting a digit from the second group to the first, and so on:

79342 15711 39387 22166

- Next the cipher clerk turns to the one-time pad. Each page of the pad contains 60 five-digit numerical groups, used as our key. The clerk takes the first group in the upper lefthand corner of the page (**26473** in this illustration) and writes it down as the first group of the message before he enciphers any text. This serves to tell the deciphering clerk on the receiving end (who has the same pad) which page to use.

26473 79342 15711 39387 22166

- Starting with the second five-digit group on the key page, the code clerk places the numbers from the pad beneath the numbers from the code book. He then adds the numbers together modulo 10.

26473 79342 15711 39387 22166
$\bigoplus$ **56328 29731 35682 23798**

26473 25660 34442 64969 45854

- At the end of the message text he enters the next five-digit group from the one-time pad after the last one actually used for encipherment (**46659**) and also enters as digits a five digit group (**23412**), of which the first three digits are a serial number for all of the messages on that circuit and the last two digits are the date of encipherment.

26473 25660 34442 64969 45854 46659 23412

- Finally, the five-digit groups are converted to five-letter groups by the code clerk using the following substitution to Latin letters:

0=>O, 1=>I, 2=>U, 3=>Z, 4=>T, 5=>R, 6=>E, 7=>W, 8=>A, 9=>P

- The Final Message is,
UETWZ UREEO ZTITU ETPEP TRART TEERP 23412.

Now let's understand the Decryption process at Moscow:-

- In Moscow the cable is determined to be KGB by an initial receiving point and then routed to the **KGB** cipher office and converted back to numerals.
- The deciphering clerk then goes to his copy of the one-time pad. If no messages have been reordered in transmission, the group **26473** should confirm that he has the proper page for decipherment. Further,if the group **46659** appears at a point that corresponds exactly to the number of cipher groups received, it confirms that all cipher groups have been received.
- The clerk then subtracts the ciphertext with obtained key, modulo 10. He then consults his code book and converts it back to the orignal message,
  "**Pilot delivered report about rockets.**"
- The message is then passed to the appropriate KGB officer at headquarters. The officer consults his records and converts the covername to the real name, **Ullmann**.

The Big Blunder

Many countries knew the benefits of One Time Pad.But in view of the daunting requirements for skilled personnel to produce huge quantities of one-time pads, each page containing a unique random-number cipher, they chose another route. The Soviets, who had a well-deserved reputation for obsession with secrecy, chose to do things the hard way, using one-time pads and paying the price of employing an army of code-makers. They capitalized on their unbreakable cipher for decades, but for a time though Soviet cryptographers were unable to produce new key pages fast enough to meet demand

- In June 1941, Adolf Hitler broke his alliance with Joseph Stalin and invaded the USSR. Overnight the production of ciphered messages by Soviet diplomatic and intelligence offices skyrocketed.
- The KGB office had strictly instructed against page duplication. But cryptologists who designed them duplicated pages to save the excess burden.
- For a period Soviet cryptographers doubled their output by duplicating cipher pages, almost certainly by the simple expedient of using extra carbon sheets in the machines that typed out the key pages.

The Soviets were no fools though,They duplicated individual pages, then shuffled them into different pads, often with different page numbers, and distributed them to Soviet offices across the globe. The US government's intelligence agency, NSA took up the challenge. It had extensive resources. Code-breakers from a variety of backgrounds: philologists and linguists with a range of foreign language skills, mathematicians to whom numerical ciphers were a technical challenge, engineers and technicians who worked on radio interception and construction of cipher machines, and English majors with a love for puzzles. Thus started **Project VENONA**.

Project VENONA

NSA and Start of VENONA

- The NSA originated as the Army's Signal Intelligence Service. It had limited capacity. In September 1939, when Germany invaded Poland, the entire staff of the Signal Intelligence Service numbered merely 19!!.
- During the war,the Army quickly added thousands of staff members to its communications intelligence and code-breaking effort, and the service's personnel grew to more than ten thousand by 1945.This included not just code-makers and code-breakers but also large radio interception staffs in Washington and in various field locations, support personnel, and the headquarters section.
- In 1952 the agency was given its current name, the National Security Agency, and came under the supervision of the secretary of defense. It was recognized as the most skilled code-breaking agency in the world.

The US primarily focused on German and Japanese ciphers during the war but nevertheless they stored the enciphered Soviet data.

- The Finns had an excellent cryptographic staff that concentrated almost entirely on the Soviet Union, the chief threat to Finland. The Finns had not broken any of the Soviet diplomatic ciphers, but they had developed a partial understanding of their external characteristics and were able to sort most diplomatic messages into a series of identifiable variants.
- Finnish-japanese information allowed American cryptanalysts to separate messages into homogeneous set. One was called Trade, because it carried the traffic of Amtorg, the Soviet trading agency.
- A breakthrough came in autumn 1943, when Lt. Richard Hallock,analyzing Trade messages, demonstrated that the Soviets were making extensive use of duplicate key pages assembled in separate one-time pad books. The work of Hallock and his colleagues opened the first crack in the walls of the Soviet code and yielded methods for discovering duplicate pages.

The Venona Project began because Carter Clarke did not trust Joseph Stalin. Colonel Clarke was chief of the U.S. Army's Special Branch, part of the War Department's Military Intelligence Division, and in 1943 its officers heard vague rumors of secret German-Soviet peace negotiations. In February 1943 Clarke ordered the secret intelligence service to establish a small program to examine ciphered Soviet diplomatic cablegrams.

Cryptanalysis

- Much of the decryption process involved **guessing** some part of the message and **recovering parts of the key**. The reconstructed key allowed decryption of other messages known to be encrypted by the same key.

- Brilliant cryptologist Samuel Chew, found out that many of the Trade messages were time-sensitive but otherwise routine notifications to Moscow that certain ships were leaving American ports with specified cargoes of Lend-Lease supplies.

- Chew's work had a dramatic impact on the efforts of those who were recovering cipher keys. They were now able to solve or remove the one-time pad cipher for stretches of Trade code text and thereby to reveal significant strings of KGB code.

While breaking the one-time pad cipher was a key to the success of Venona, it was not the end of the code-breaking project, because the Soviets used a two-stage system. The Soviet code clerk first encoded the message from a code book and then further complicated the code by use of the one-time pad cipher. The duplicated cipher pages allowed the NSA to break the second part, but not the first. To solve the first part and make the message readable, crypto linguists had to recreate the Soviet code book.

- Finnish troops had captured the code book when they overran the Soviet consulate at Pesarno, Finland, in June 1941.17 The Germans then obtained the book from the Finns. In May 1945 a U.S. Army intelligence team retrieved a copy at a German signals intelligence archive located in a castle in Saxony, Germany.

Aftermath

In Venona Project more than twenty-nine hundred messages amounting to more than five thousand pages of text were decrypted and read. But as impressive as that feat was, the messages represented only a fraction of the total Soviet cable traffic. Venona uncovered, in whole or in part, roughly half (**49%**) of the messages sent in 1944 between the KGB New York office and its Moscow headquarters, but only **15%** of the messages from 1943 and a mere 1.8% of the messages from 1942 (only twenty-three out of nearly thirteen hundred). Only **1.5%** of the 1945 traffic between the KGB Washington office and Moscow was deciphered.

The Soviets believed that the real security of their cables lay in their use of the unbreakable one-time pad cipher, not in the lesser security device of cover names. Cover names frequently hinted at a person's profession or character-a serious error if security is a primary consideration. George Silverman, a Soviet source in the U.S. Army Air Force, had the cover name **Aileron**. Harold Glasser, a Soviet agent in the U.S. Treasury, received the cover name **Ruble**, the name of the Soviet currency.The KGB gave Trotskyists and Zionists, both hated enemies, the cover names **Polecats** and **Rats**, respectively.
KAPITAN (President Roosevelt), **COUNTRY** (the United States), **BANK** (United States State Department), **CARTHAGE** (Washington, D.C.), **BABYLON** (San Francisco), **ARSENAL** (United States War Department) were several such other code names.

Refrences and Bonus

Nuclear War Impact

- Deciphered messages led the FBI to identify Klaus Fuchs, a British physicist working on the atomic bomb project, as a spy, a discovery that ultimately brought about the arrests of Julius and Ethel Rosenberg on charges of espionage for passing atomic secrets to the Soviet Union.
- These 49 messages detailed Soviet atomic espionage and unmistakably showed that Julius Rosenberg was a Soviet agent, an assertion that some historians and much of the American public had rejected as unsupported by documentary evidence.
- The betrayal of American atomic secrets to the Soviets allowed the Soviet Union to develop atomic weapons several years sooner and at a substantially lower cost than it otherwise would have.
- Several big events in modern History like Cuban Missile Crisis, US's withdrawal from Korea and most significantly the Atomic Arms race that shaped the true horrors of cold war.

Joseph Stalin's knowledge that espionage assured the Soviet Union of quickly breaking the American atomic monopoly emboldened his diplomatic strategy in his early Cold War clashes with the United States. It is doubtful that Stalin, rarely a risk-taker, would have supplied the military wherewithal and authorized North Korea to invade South Korea in 1950 had the Soviet Union not exploded an atomic bomb in 1949. Otherwise Stalin might have feared that President Harry Truman would stanch any North Korean invasion by threatening to use atomic weapons. After all,as soon as the atomic bomb had been developed, Truman had not hesitated to use it twice to end the war with Japan. But in 1950, with Stalin in possession of the atomic bomb, Truman was deterred from using atomic weapons in Korea, even in the late summer when initially unprepared American forces were driven back into the tip of Korea and in danger of being pushed into the sea, and then again in the winter when Communist Chinese forces entered the war.

References

1. VENONA: Decoding Soviet Espionage in America, by John Earl Haynes and Harvey Klehr.
2. Venona: Soviet Espionage and the American Response, 1939-1957 by NSA.
3. Project VENONA: Breaking the Unbreakable Code by Cassandra Hankin.