BILL, RECORD LECTURE!!!!

BILL RECORD LECTURE!!!
Something Wrong With All Ciphers So Far. Fix it with Randomization

June 26, 2021
Eve CAN tell... 

Let $C$ be any of Shift, Affine, Gen Sub, Vig, Matrix, (NOT one-time pad, Book-Vig, Autokey-Vig, Book-Cipher). Assume Eve does not know how to crack $C$. 

But Eve can still tell if two messages are the same or not. EASILY! 

Is this a problem? YES! 

Eve knows that the message will say where the spy is. The message will be of the form a city and then a state, so for example IthacaNewYork Alice sends to Bob adecn aapad ecnaa p. 

Eve notices adecnaap adecnaap. Eve knows that the city and state are the same!
Eve CAN tell...

Let C be any of Shift, Affine, Gen Sub, Vig, Matrix, (NOT one-time pad, Book-Vig, Autokey-Vig, Book-Cipher). Assume Eve does not know how to crack C. **But** Eve can still tell if two messages are the same or not. EASILY!
Is this a problem?
Eve CAN tell...

Let $C$ be any of Shift, Affine, Gen Sub, Vig, Matrix, (NOT one-time pad, Book-Vig, Autokey-Vig, Book-Cipher). Assume Eve does not know how to crack $C$. But Eve can still tell if two messages are the same or not. EASILY!

Is this a problem?

YES! Eve knows that the message will say where the spy is. The message will be of the form a city and then a state, so for example

IthacaNewYork
Let $C$ be any of Shift, Affine, Gen Sub, Vig, Matrix, (NOT one-time pad, Book-Vig, Autokey-Vig, Book-Cipher). Assume Eve does not know how to crack $C$. But Eve can still tell if two messages are the same or not. EASILY!

Is this a problem?

YES! Eve knows that the message will say where the spy is. The message will be of the form a city and then a state, so for example IthacaNewYork

Alice sends to Bob adecn aapad ecnaa p.
Eve CAN tell...

Let $C$ be any of Shift, Affine, Gen Sub, Vig, Matrix, (NOT one-time pad, Book-Vig, Autokey-Vig, Book-Cipher). Assume Eve does not know how to crack $C$.

But Eve can still tell if two messages are the same or not. EASILY!

Is this a problem?

YES! Eve knows that the message will say where the spy is. The message will be of the form a city and then a state, so for example

\textbf{IthacaNewYork}

Alice sends to Bob \texttt{adecn aapad ecnaa p}.

Eve notices \texttt{adecnaap adecnaap}. 
Eve CAN tell . . .

Let $C$ be any of Shift, Affine, Gen Sub, Vig, Matrix, (NOT one-time pad, Book-Vig, Autokey-Vig, Book-Cipher). Assume Eve does not know how to crack $C$.

**But** Eve can still tell if two messages are **the same** or **not**. EASILY!

Is this a problem?

**YES!** Eve knows that the message will say where the spy is. The message will be of the form a city and then a state, so for example

$IthacaNewYork$

Alice sends to Bob $adecn$ $aapad$ $ecnaa$ $p$.

Eve notices $adecnaap$ $adecnaap$.

Eve knows that the city and state are the same!
What Does Eve Know?

Cities with a state’s name. * means no longer a city.
What Does Eve Know?

Cities with a state’s name. * means no longer a city.

What Does Eve Know?

Cities with a state’s name. * means no longer a city.


There are 33 such cities, 22 of which still exist. Eve’s search for the spy is reduced!
The problem of the same message leading to the same ciphertext is called

**The NY, NY Problem.**
Problem If $C$ is any of the ciphers discussed (except 1-time pad, Book-Vig) then Eve can tell when two messages are the same.

Discuss Is there a cipher for which Eve cannot tell this?
Problem If $C$ is any of the ciphers discussed (except 1-time pad, Book-Vig) then Eve can tell when two messages are the same.

Discuss Is there a cipher for which Eve cannot tell this? Need that even if $x = y$ could have $C(x) \neq C(y)$.

Discuss How can we do that?
**Problem** If $C$ is any of the ciphers discussed (except 1-time pad, Book-Vig) then Eve can tell when two messages are the same.

**Discuss** Is there a cipher for which Eve cannot tell this? Need that even if $x = y$ could have $C(x) \neq C(y)$.

**Discuss** How can we do that?

Use a very long key and keep using different parts of it, which is the 1-time pad, Book-Vig. Is there an easier way?
How to Fix the NY,NY Problem

**Problem** If $C$ is any of the ciphers discussed (except 1-time pad, Book-Vig) then Eve can tell when two messages are the same.

**Discuss** Is there a cipher for which Eve cannot tell this? Need that even if $x = y$ could have $C(x) \neq C(y)$.

**Discuss** How can we do that?

Use a very long key and keep using different parts of it, which is the 1-time pad, Book-Vig. Is there an easier way?

**Discuss** Can we do this without a long key?
How to Fix This Without a Long Key

**Obstacle** All of our ciphers are deterministic. Need Rand.
Obstacle All of our ciphers are deterministic. Need Rand.

Recall Deterministic Shift Key is $s \in S$. Math is mod 26.

1. To send message $(m_1, \ldots, m_L)$ send $(m_1 + s, \ldots, m_L + s)$.
2. To decode message $(c_1, \ldots, c_L)$ find $(c_1 - s, \ldots, c_L - s)$. 
How to Fix This Without a Long Key

**Obstacle** All of our ciphers are deterministic. Need Rand.

**Recall Deterministic Shift** Key is $s \in S$. Math is mod 26.

1. To send message $(m_1, \ldots, m_L)$ send $(m_1 + s, \ldots, m_L + s)$.
2. To decode message $(c_1, \ldots, c_L)$ find $(c_1 - s, \ldots, c_L - s)$.

**Randomized Shift** Key is a function $f : S \rightarrow S$.

1. To send message $(m_1, \ldots, m_L)$ (each $m_i$ is a character):
   1.1 Pick random $r_1, \ldots, r_L \in S$.
   1.2 Send $((r_1; m_1 + f(r_1)), \ldots, (r_L; m_L + f(r_L)))$. 

How to Fix This Without a Long Key

**Obstacle** All of our ciphers are deterministic. Need Rand.

**Recall Deterministic Shift** Key is $s \in S$. Math is mod 26.

1. To send message $(m_1, \ldots, m_L)$ send $(m_1 + s, \ldots, m_L + s)$.
2. To decode message $(c_1, \ldots, c_L)$ find $(c_1 - s, \ldots, c_L - s)$.

**Randomized Shift** Key is a function $f : S \rightarrow S$.

1. To send message $(m_1, \ldots, m_L)$ (each $m_i$ is a character):
   1.1 Pick random $r_1, \ldots, r_L \in S$.
   1.2 Send $((r_1; m_1 + f(r_1)), \ldots, (r_L; m_L + f(r_L)))$.
2. To decode message $((r_1; c_1), \ldots, (r_L; c_L))$:
   2.1 Find $(c_1 - f(r_1), \ldots, c_L - f(r_L))$. 
Example

The key is $f(r) = 2r + 7$. Alice wants to send NY,NY which we interpret as nyny. Need four shifts.

Pick random $r = 4$, so first shift is $2 \times 4 + 7 = 15$
Pick random $r = 10$, so second shift is $2 \times 10 + 7 = 1$
Pick random $r = 1$, so third shift is $2 \times 1 + 7 = 9$
Pick random $r = 17$, so fourth shift is $2 \times 17 + 7 = 15$

Send (4;C), (10;Z), (1;W), (17;N)

Eve will not be able to tell that is of the form XYXY.
PROS and CONS of Randomized Shift

Discuss

If Alice sends NY, NY, Eve can't tell its XYXY.

PRO

Generally, Eve cannot tell if 2 messages are same.

CON

More effort on Alice and Bob's part.

Question

Is Randomized Shift crackable? Discuss.
PROS and CONS of Randomized Shift

Discuss

**PRO** If Alice sends **NY,NY** Eve can’t tell its **XYXY**.
Discuss

**PRO** If Alice sends **NY,NY** Eve can’t tell its **XYXY**.

**PRO** Generally, Eve cannot tell if 2 messages are same.
Discuss

**PRO** If Alice sends **NY,NY** Eve can’t tell its **XYXY**.

**PRO** Generally, Eve cannot tell if 2 messages are same.

**CON** More effort on Alice and Bob’s part.
Discuss

**PRO** If Alice sends **NY, NY** Eve can’t tell its **XYXY**.

**PRO** Generally, Eve cannot tell if 2 messages are same.

**CON** More effort on Alice and Bob’s part.

**Question** Is Randomized Shift crackable? Discuss.
Cracking Randomized Shift

June 26, 2021
Cracking Randomized Shift

With a long text Rand Shift is crackable. If $N$ is long and Eve sees:

$$(r_1; \sigma_1)(r_2; \sigma_2) \cdots (r_N; \sigma_N).$$

View as:

1. There are only 26 possible $r$.
2. There are $N$ pairs of the form $(r_i, \sigma_i)$.
3. Some $r$ appears $N/26$ times by PHP (Pigeon Hole Princ).

So have, with $L = \frac{N}{26}$:

$$(r; \sigma_{i_1}) \cdots (r; \sigma_{i_2}) \cdots \cdots (r; \sigma_{i_L})$$
Cracking Randomized Shift (cont)

So we have:

\[(r; \sigma_{i_1}) \cdots (r; \sigma_{i_2}) \cdots \cdots (r; \sigma_{i_L})\]

where \(L\) is large.

So \(\sigma_{i_1}, \ldots, \sigma_{i_L}\) are all coded by the same shift.
So we have:

\[(r; \sigma_{i_1}) \cdots (r; \sigma_{i_2}) \cdots \cdots (r; \sigma_{i_L})\]

where \(L\) is large.

So \(\sigma_{i_1}, \ldots, \sigma_{i_L}\) are all coded by the same shift.

1. From our study of Vig we know that taking every \(m\)th letter in a text has the same distribution of letters as a normal text.
So we have:

\[(r; \sigma_{i_1}) \cdots (r; \sigma_{i_2}) \cdots \cdots (r; \sigma_{i_L})\]

where \(L\) is large.

So \(\sigma_{i_1}, \ldots, \sigma_{i_L}\) are all coded by the same shift.

1. From our study of Vig we know that taking every \(m\)th letter in a text has the same distribution of letters as a normal text.

2. It turns out that taking a random set of letters also has the same distribution as a normal text.
So we have:

\[(r; \sigma_{i_1}) \cdots (r; \sigma_{i_2}) \cdots \cdots (r; \sigma_{i_L})\]

where \(L\) is large.
So \(\sigma_{i_1}, \ldots, \sigma_{i_L}\) are all coded by the same shift.

1. From our study of Vig we know that taking every \(m\)th letter in a text has the same distribution of letters as a normal text.
2. It turns out that taking a random set of letters also has the same distribution as a normal text.

**Good News** Try all shifts and use **Is English**.
So we have:

\[(r; \sigma_{i_1}) \cdots (r; \sigma_{i_2}) \cdots \cdots (r; \sigma_{i_L})\]

where \(L\) is large.

So \(\sigma_{i_1}, \ldots, \sigma_{i_L}\) are all coded by the same shift.

1. From our study of Vig we know that taking every \(m\)th letter in a text has the same distribution of letters as a normal text.

2. It turns out that taking a random set of letters also has the same distribution as a normal text.

**Good News** Try all shifts and use Is English.

**Bad News** Just tells us which shift this particular \(r\) maps to.
So we have:

\[(r; \sigma_{i_1}) \cdots (r; \sigma_{i_2}) \cdots \cdots (r; \sigma_{i_L})\]

where \( L \) is large.

So \( \sigma_{i_1}, \ldots, \sigma_{i_L} \) are all coded by the same shift.

1. From our study of Vig we know that taking every \( m \)th letter in a text has the same distribution of letters as a normal text.

2. It turns out that taking a random set of letters also has the same distribution as a normal text.

**Good News** Try all shifts and use **Is English**.

**Bad News** Just tells us which shift this particular \( r \) maps to.

Next Slide deals with this.
Many $r$ Will Appear Many Times

Recall the following reasoning:

$$(r_1, \sigma_1)(r_2, \sigma_2) \cdots (r_N, \sigma_N)$$

View as:

1. There are only 26 possible $r$.
2. There are $N$ pairs of the form $(r_i, \sigma_i)$.
3. Some $r$ appears $N/26$ times by Pigeon Hole Principle.
Many $r$ Will Appear Many Times

Recall the following reasoning:

$$(r_1; \sigma_1)(r_2; \sigma_2) \cdots (r_N; \sigma_N)$$

View as:

1. There are only 26 possible $r$.
2. There are $N$ pairs of the form $(r_i, \sigma_i)$.
3. Some $r$ appears $N/26$ times by Pigeon Hole Principle.

We can do better.
Many $r$ Will Appear Many Times

Recall the following reasoning:

$$(r_1; \sigma_1)(r_2; \sigma_2) \cdots (r_N; \sigma_N)$$

View as:

1. There are only 26 possible $r$.
2. There are $N$ pairs of the form $(r_i, \sigma_i)$.
3. Some $r$ appears $N/26$ times by Pigeon Hole Principle.

We can do better. The $r$’s are picked unif at random.
Eve’s Math Problem

Eve sees

$$(r_1, \sigma_1), (r_2, \sigma_2), \ldots, (r_N, \sigma_N)$$

Want that ALL $r$’s appear LOTS of times.
Eve’s Math Problem

Eve sees

\[(r_1, \sigma_1), (r_2, \sigma_2), \ldots, (r_N, \sigma_N)\]

Want that ALL \(r\)'s appear LOTS of times.

**Wrong Question**
Eve’s Math Problem

Eve sees

\[(r_1, \sigma_1), (r_2, \sigma_2), \cdots, (r_N, \sigma_N)\]

Want that ALL \(r\)’s appear LOTS of times.

**Wrong Question**

Eve sees

\[(r_1, \sigma_1), (r_2, \sigma_2), \cdots, (r_N, \sigma_N)\]

The \(r_i\)’s are picked uniformly from \(\{0, \ldots, 25\}\).
Eve’s Math Problem

Eve sees

\[(r_1, \sigma_1), (r_2, \sigma_2), \cdots, (r_N, \sigma_N)\]

Want that ALL \(r\)'s appear LOTS of times.

**Wrong Question**

Eve sees

\[(r_1, \sigma_1), (r_2, \sigma_2), \cdots, (r_N, \sigma_N)\]

The \(r_i\)'s are picked uniformly from \(\{0, \ldots, 25\}\).

Want the prob that MOST \(r\) appear ALOT of times is large.
Chebyshev’s Inequality

We put $N$ balls into $n$ bins uniformly at random. $N$ big, $n$ small.
Chebyshev’s Inequality

We put $N$ balls into $n$ bins uniformly at random. $N$ big, $n$ small. Let $X_r$ be the number of balls in bin $r$. 

Chebyshev’s Inequality

If $X$ is a random variable then 

$$\Pr\left( |X - \mathbb{E}(X)| \geq k\sigma \right) \leq \frac{1}{k^2}$$

where $\sigma = \sqrt{\text{Var}(X)}$, the Variance of $X$.

Using this we find that for our problem: 

$$\Pr\left( \text{all } r \in \{1, \ldots, 26\} \text{ appear } \geq \frac{N}{260} \text{ times} \right) \geq 0.999999999$$

Hence can find, for all $r$, what shift $r$ maps to.
Chebyshev’s Inequality

We put $N$ balls into $n$ bins uniformly at random. $N$ big, $n$ small. Let $X_r$ be the number of balls in bin $r$. The expected value of $X_r$, denoted $E(X_r)$ is $\frac{N}{n}$.
Chebyshev’s Inequality

We put $N$ balls into $n$ bins uniformly at random. $N$ big, $n$ small.
Let $X_r$ be the number of balls in bin $r$.
The expected value of $X_r$, denoted $E(X_r)$ is $\frac{N}{n}$.
What is the probability that $X_r$ will be much lower than $\frac{N}{n}$?
Chebyshev’s Inequality

We put $N$ balls into $n$ bins uniformly at random. $N$ big, $n$ small.
Let $X_r$ be the number of balls in bin $r$.
The expected value of $X_r$, denoted $E(X_r)$ is $\frac{N}{n}$.
What is the probability that $X_r$ will be much lower than $\frac{N}{n}$?
We won’t answer that, but we will say how to answer it:

**Chebyshev’s Inequality** If $X$ is a random variable then

$$\Pr(|X - E(X)| \geq k\sigma) \leq \frac{1}{k^2}$$

where $\sigma = \sqrt{V(X)}$, the Variance of $X$. 
Chebyshev’s Inequality

We put $N$ balls into $n$ bins uniformly at random. $N$ big, $n$ small.
Let $X_r$ be the number of balls in bin $r$.
The expected value of $X_r$, denoted $E(X_r)$ is $\frac{N}{n}$.
What is the probability that $X_r$ will be much lower than $\frac{N}{n}$?
We won’t answer that, but we will say how to answer it:
**Chebyshev’s Inequality** If $X$ is a random variable then

$$
\Pr(|X - E(X)| \geq k\sigma) \leq \frac{1}{k^2}
$$

where $\sigma = \sqrt{\text{Var}(X)}$, the Variance of $X$.
Using this we find that for our problem:
Chebyshev’s Inequality

We put $N$ balls into $n$ bins uniformly at random. $N$ big, $n$ small. Let $X_r$ be the number of balls in bin $r$.

The expected value of $X_r$, denoted $E(X_r)$ is $\frac{N}{n}$.

What is the probability that $X_r$ will be much lower than $\frac{N}{n}$?

We won’t answer that, but we will say how to answer it:

**Chebyshev’s Inequality** If $X$ is a random variable then

$$\Pr(|X - E(X)| \geq k\sigma) \leq \frac{1}{k^2}$$

where $\sigma = \sqrt{V(X)}$, the Variance of $X$.

Using this we find that for our problem:

$$\Pr(\text{all } r \in \{1, \ldots, 26\} \text{ appear } \geq \frac{N}{260} \text{ times}) \geq 0.999999999$$
Chebyshev’s Inequality

We put $N$ balls into $n$ bins uniformly at random. $N$ big, $n$ small. Let $X_r$ be the number of balls in bin $r$. The expected value of $X_r$, denoted $E(X_r)$ is $\frac{N}{n}$. What is the probability that $X_r$ will be much lower than $\frac{N}{n}$?

We won’t answer that, but we will say how to answer it: **Chebyshev’s Inequality**

If $X$ is a random variable then

$$\Pr(|X - E(X)| \geq k\sigma) \leq \frac{1}{k^2}$$

where $\sigma = \sqrt{V(X)}$, the Variance of $X$.

Using this we find that for our problem:

$$\Pr(\text{all } r \in \{1, \ldots, 26\} \text{ appear } \geq \frac{N}{260} \text{ times}) \geq 0.9999999999$$

Hence can find, for all $r$, what shift $r$ maps to.
Cracking Randomized Shift Final Algorithm

1. Input \((r_1, \sigma_1)\), \((r_2, \sigma_2)\), \(\ldots\), \((r_N, \sigma_N)\).

2. For each \(r \in \{1, \ldots, 26\}\):
   2.1 Look at the spots \((r, \sigma_1)\), \((r, \sigma_2)\), \(\ldots\), \((r, \sigma_L)\).
   2.2 All of these \(\sigma_i\)'s used same shift.
   2.3 Guess each shift and use IS-ENGLISH to find out which shift is correct.

3. We now have the mapping of \(r\)'s to shifts. \(r\) maps to shift \(s_r\).

4. Can use the \(s_r\)'s to decode entire message.
Cracking Randomized Shift Final Algorithm

1. Input \((r_1; \sigma_1)(r_2; \sigma_2) \cdots (r_N; \sigma_N)\)
Cracking Randomized Shift Final Algorithm

1. Input \((r_1; \sigma_1)(r_2; \sigma_2) \cdots (r_N; \sigma_N)\)
2. For each \(r \in \{1, \ldots, 26\}:\)
1. Input \((r_1; \sigma_1)(r_2; \sigma_2) \cdots (r_N; \sigma_N)\)

2. For each \(r \in \{1, \ldots, 26\}\):
   
   2.1 Look at the spots \((r, \sigma)\), so

\[(r, \sigma_1) \cdots (r, \sigma_2) \cdots (r, \sigma_L).\]
1. Input \((r_1; \sigma_1)(r_2; \sigma_2) \cdots (r_N; \sigma_N)\)

2. For each \(r \in \{1, \ldots, 26\}\):
   2.1 Look at the spots \((r, \sigma)\), so
   \[
   (r, \sigma_1) \cdots (r, \sigma_2) \cdots (r, \sigma_L).
   \]

2.2 All of these \(\sigma_{ij}\)'s used same shift.

3. We now have the mapping of \(r\)'s to shifts. \(r\) maps to shift \(s_r\).

4. Can use the \(s_r\)'s to decode entire message.
Cracking Randomized Shift Final Algorithm

1. Input \((r_1; \sigma_1)(r_2; \sigma_2) \cdots (r_N; \sigma_N)\)
2. For each \(r \in \{1, \ldots, 26\}\):
   2.1 Look at the spots \((r, \sigma)\), so
   \[(r, \sigma_1) \cdots (r, \sigma_2) \cdots (r, \sigma_L).\]
   2.2 All of these \(\sigma_i\)'s used same shift.
   2.3 Guess each shift and use IS-ENGLISH to find out which shift is correct.
3. We now have the mapping of \(r\)'s to shifts. \(r\) maps to shift \(s_r\).
4. Can use the \(s_r\)'s to decode entire message.
Cracking Randomized Shift Final Algorithm

1. Input \((r_1; \sigma_1)(r_2; \sigma_2) \cdots (r_N; \sigma_N)\)
2. For each \(r \in \{1, \ldots, 26\}\):
   2.1 Look at the spots \((r, \sigma)\), so
      \[(r, \sigma_1) \cdots (r, \sigma_2) \cdots (r, \sigma_L)\].
   2.2 All of these \(\sigma_{ij}\)'s used same shift.
   2.3 Guess each shift and use IS-ENGLISH to find out which shift is correct.
3. We now have the mapping of \(r\)'s to shifts. \(r\) maps to shift \(s_r\).
1. Input \((r_1; \sigma_1)(r_2; \sigma_2) \cdots (r_N; \sigma_N)\)

2. For each \(r \in \{1, \ldots, 26\}\):
   2.1 Look at the spots \((r, \sigma),\) so
   \[(r, \sigma_1) \cdots (r, \sigma_2) \cdots (r, \sigma_L).\]
   2.2 All of these \(\sigma_i\)'s used same shift.
   2.3 Guess each shift and use IS-ENGLISH to find out which shift is correct.

3. We now have the mapping of \(r\)'s to shifts. \(r\) maps to shift \(s_r\).

4. Can use the \(s_r\)'s to decode entire message.
History And Uses of the Randomized Shift

The **Randomized Shift** was invented in
History And Uses of the Randomized Shift

The **Randomized Shift** was invented in 2019 by William Gasarch while preparing to teach CMSC/MATH/ENEE 456.
History And Uses of the Randomized Shift

The **Randomized Shift** was invented in 2019 by William Gasarch while preparing to teach CMSC/MATH/ENEE 456.

1. It has never been used.
History And Uses of the Randomized Shift

The Randomized Shift was invented in 2019 by William Gasarch while preparing to teach CMSC/MATH/ENEE 456.

1. It has never been used.
2. The general technique of adding randomness to a known cipher to avoid the NY,NY problem is used all of the time.
The **Randomized Shift** was invented in 2019 by William Gasarch while preparing to teach CMSC/MATH/ENEE 456.

1. It has never been used.
2. The general technique of adding randomness to a known cipher to avoid the NY,NY problem is used all of the time.
3. The terminology **NY,NY** problem and the example we gave are also due to me.
History And Uses of the Randomized Shift

The **Randomized Shift** was invented in 2019 by William Gasarch while preparing to teach CMSC/MATH/ENEE 456.

1. It has never been used.
2. The general technique of adding randomness to a known cipher to avoid the NY,NY problem is used all of the time.
3. The terminology **NY,NY** problem and the example we gave are also due to me.
4. I am telling you this to warn you that if you are on a job interview with the NSA and you say *I learned to use the randomized shift to solve the NY,NY problem* they will not know what you are talking about.

Boo!

2. We can turn any Det. Cipher into a randomized one. Will use this later in the course.

3. If turn a weak Det. Cipher (like Shift) into a randomized one, still crackable.

4. Cracking it takes a much longer text.
Upshot


2. We can turn any Det. Cipher into a randomized one. Will use this later in the course.

2. We can turn any Det. Cipher into a randomized one. Will use this later in the course.

3. If turn a weak Det. Cipher (like Shift) into a randomized one, still crackable.

2. We can turn any Det. Cipher into a randomized one. Will use this later in the course.

3. If turn a weak Det. Cipher (like Shift) into a randomized one, still crackable.

4. Cracking it takes a much longer text.
BILL, STOP RECORDING LECTURE!!!!

BILL STOP RECORDING LECTURE!!!!