Modern Cryptography

Notes on August 27, 2021 reading

Jeffrey Goldberg
jeffrey@goldmark.org

August 27, 2021
Revised May 8, 2024

Modern Cryptography

Modern cryptography is a combination of

- Well-defined security goals (such as indistinguishability)
- Well-defined attacker models (in terms of “games”)
- Rigorous proof strategies (typically reductions)

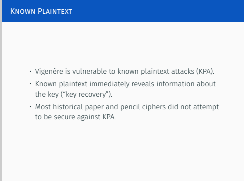
Review

Secrecy isn't all

- The One Time Pad (OTP) offers perfect secrecy.
- The OTP is malleable and so, among other things, is not suitable for sending instructions.
- Many schemes that *approximate* a One Time Pad do not provide approximate perfect secrecy. Small deviations can produce *catastrophic* failures.

Known Plaintext

- Vigenère is vulnerable to known plaintext attacks (KPA).
- Known plaintext immediately reveals information about the key (“key recovery”).
- Most historical paper and pencil ciphers did not attempt to be secure against KPA.



Known Plaintext

- Vigenère is vulnerable to known plaintext attacks (KPA).
- Known plaintext immediately reveals information about the key (“key recovery”).
- Most historical paper and pencil ciphers did not attempt to be secure against KPA.

1. That last point wasn’t from last week, but it is still where it belongs on this slide.

Distinguishing notions

Distinguishing notions

IND-CPA

Secrecy: The concept

Definition 1 (Secrecy)

A cipher scheme provides secrecy if and only if the ciphertext provides no new information in narrowing down what the plaintext is.

Secrecy: Unconditional probability

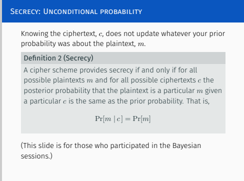
Knowing the ciphertext, c, does not update whatever your prior probability was about the plaintext, m.

Definition 2 (Secrecy)

A cipher scheme provides secrecy if and only if for all possible plaintexts m and for all possible ciphertexts c the posterior probability that the plaintext is a particular m given a particular c is the same as the prior probability. That is,

$$\Pr[m \mid c] = \Pr[m]$$

(This slide is for those who participated in the Bayesian sessions.)



Secrecy: Unconditional probability

Knowing the ciphertext, c, does not update whatever your prior probability was about the plaintext, m.

Definition 2 (Secrecy)

A cipher scheme provides secrecy if and only if for all possible plaintexts m and for all possible ciphertexts c the posterior probability that the plaintext is a particular m given a particular c is the same as the prior probability. That is,

$$\Pr[m \mid c] = \Pr[m]$$

(This slide is for those who participated in the Bayesian sessions.)

1. Why am I not using the same notation as in *Serious Cryptography*? I could spin a story about wanting to make people more familiar with more common notation. And that wouldn't be a lie. But there is also a nice LaTeX package for cryptographic notation, and it is easier to stick with its defaults.
2. Despite the formalism this isn't the formal definition, which has lots of cruft and restrictions pin things down like lengths of messages, etc.
3. If this isn't helpful, just ignore this slide

IND-CPA& IND-EAV

Indistinguishability in the presence of an eavesdropper, IND-EAV, is a weaker security notion than indistinguishability in the presence of a chosen plaintext attack, IND-CPA. But the formal definition of IND-EAV is easier to present then the formal definition of IND-CPA. So I may switch back and forth.

IND-CPA: The concept

If there is no discernible relationship between the plaintext, m, and the ciphertext, c, then the adversary, $\mathcal{A}$, must not have any advantage[1] in determining which plaintext corresponds to which ciphertext.

[1] By “advantage” we mean determining correctly at a rate better than chance.

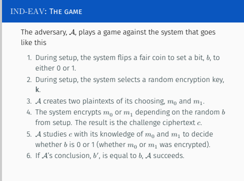
Secrecy is IND-CPA (and IND-CPA is secrecy)

When both secrecy and IND-CPA are formally defined, they can be shown to be equivalent. But the IND-CPA game definition turns out to be more useful and adaptable to other security notions and for constructing proofs about specific encryption schemes.

IND-EAV: The game

The adversary, $\mathcal{A}$, plays a game against the system that goes like this

1. During setup, the system flips a fair coin to set a bit, b, to either 0 or 1.
2. During setup, the system selects a random encryption key, k.
3. $\mathcal{A}$ creates two plaintexts of its choosing, m_0 and m_1.
4. The system encrypts m_0 or m_1 depending on the random b from setup. The result is the challenge ciphertext c.
5. $\mathcal{A}$ studies c with its knowledge of m_0 and m_1 to decide whether b is 0 or 1 (whether m_0 or m_1 was encrypted).
6. If $\mathcal{A}$'s conclusion, b', is equal to b, $\mathcal{A}$ succeeds.



IND-EAV: The game

The adversary, $\mathcal{A}$, plays a game against the system that goes like this

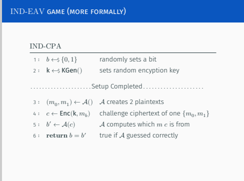
1. During setup, the system flips a fair coin to set a bit, b, to either 0 or 1.
2. During setup, the system selects a random encryption key, **k**.
3. $\mathcal{A}$ creates two plaintexts of its choosing, m_0 and m_1.
4. The system encrypts m_0 or m_1 depending on the random b from setup. The result is the challenge ciphertext c.
5. $\mathcal{A}$ studies c with its knowledge of m_0 and m_1 to decide whether b is 0 or 1 (whether m_0 or m_1 was encrypted).
6. If $\mathcal{A}$'s conclusion, b', is equal to b, $\mathcal{A}$ succeeds.

1. This is the IND-EAV game. The real IND-CPA game gives $\mathcal{A}$ the ability to create many pairs. But if someting fails IND-EAV it will also fail IND-CPA.

IND-EAV game (more formally)

IND-CPA

```
$b \leftarrow\$ \{0,1\}$                     randomly sets a bit
$\mathsf{k} \leftarrow\$ \mathsf{KGen}()$    sets random encyption key
.......................Setup Completed.......................
$(m_0, m_1) \leftarrow \mathcal{A}()$        $\mathcal{A}$ creates 2 plaintexts
$c \leftarrow \mathsf{Enc}(\mathsf{k}, m_b)$ challenge ciphertext of one $\{m_0, m_1\}$
$b' \leftarrow \mathcal{A}(c)$               $\mathcal{A}$ computes which $m$ $c$ is from
$\mathbf{return}\ b = b'$                   true if $\mathcal{A}$ guessed correctly
```



IND-EAV game (more formally)

IND-CPA

```
$b \leftarrow\$ \{0,1\}$                 randomly sets a bit
$\mathsf{k} \leftarrow\$ \mathsf{KGen}()$            sets random encyption key
.....................Setup Completed.....................
$(m_0, m_1) \leftarrow \mathcal{A}()$    $\mathcal{A}$ creates 2 plaintexts
$c \leftarrow \mathsf{Enc}(\mathsf{k}, m_b)$      challenge ciphertext of one $\{m_0, m_1\}$
$b' \leftarrow \mathcal{A}(c)$           $\mathcal{A}$ computes which $m$ $c$ is from
$\mathbf{return}\ b = b'$          true if $\mathcal{A}$ guessed correctly
```

The notation “$x \leftarrow\$ f()$” to indicate that a random value is returned comes from using a currency symbol to represent flipping coins.

Enigma Example

Enigma: Rise of the machines

Enigma-like ciphers were

- Designed to secure against known plaintext attacks (KPA);
- Designed to remain secure even if adversary captured a device; (Kerckhoffs's Principle);
- But Germans still made great efforts keep the system secret.

Enigma: Fall of the machines

- Enigma was not fully secure ciphertext only (COA):
    - Reflector made easy distinguishability.
    - Marian Rejewski discovered a partial key recovery (which rotors in which order) ciphertext only attack (COA) in 1932.
- Without capturing a device, Rejewski and his team were able to completely reverse engineer the system from known plaintexts. (Illustrating the importance of Kerckhoffs's Principle)
- Other KPAs were found by the Polish Cipher Bureau in following years.

Those breaks were not sufficient to decrypt any messages, but they were foundational to what would follow in the UK.

Enigma Example

Exploiting the reflector

Reflection never mirrors

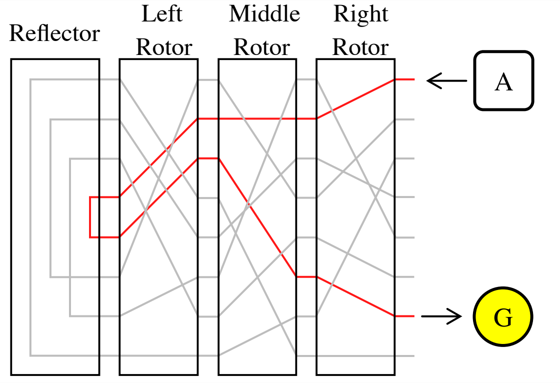


Figure 1: Enigma rotors and reflector: Reflector can never send signal back the way it came, so no letter ever encrypts to itself. Source

$\mathcal{A}$ versus Enigma: An IND-CPA game example

A characteristic of most Enigma-like systems is that no letter could ever encrypt to itself. The $\mathcal{A}$ will use that to its advantage.

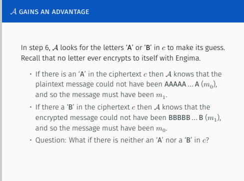
IND-CPA against **EEnigma**

```
$m_0 \leftarrow \mathsf{AAAAA \ldots A}$                 $\mathcal{A}$ sets $m_0$ to long string of As
$m_1 \leftarrow \mathsf{BBBBB \ldots B}$                 $\mathcal{A}$ sets $m_1$ to long string of Bs
$c \leftarrow \mathsf{EEnigma}(\mathsf{k}, m_b)$         gets ciphertext of one
$b' \leftarrow \mathcal{A}(c)$                           $\mathcal{A}$ “guesses” which $m$
$\mathbf{return}\ b = b'$                                true if $\mathcal{A}$ guessed correctly
```

$\mathcal{A}$ GAINS AN ADVANTAGE

In step 6, $\mathcal{A}$ looks for the letters 'A' or 'B' in c to make its guess. Recall that no letter ever encrypts to itself with Engima.

- If there is an 'A' in the ciphertext c then $\mathcal{A}$ knows that the plaintext message could not have been AAAAA … A (m_0), and so the message must have been m_1.
- If there a 'B' in the ciphertext c then $\mathcal{A}$ knows that the encrypted message could not have been BBBBB … B (m_1), and so the message must have been m_0.
- Question: What if there is neither an 'A' nor a 'B' in c?



$\mathcal{A}$ GAINS AN ADVANTAGE

In step 6, $\mathcal{A}$ looks for the letters ‘A’ or ‘B’ in c to make its guess. Recall that no letter ever encrypts to itself with Engima.

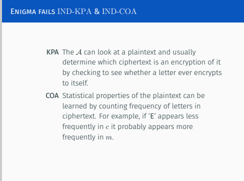
- If there is an ‘A’ in the ciphertext c then $\mathcal{A}$ knows that the plaintext message could not have been AAAAA … A (m_0), and so the message must have been m_1.
- If there a ‘B’ in the ciphertext c then $\mathcal{A}$ knows that the encrypted message could not have been BBBBB … B (m_1), and so the message must have been m_0.
- Question: What if there is neither an ‘A’ nor a ‘B’ in c?

1. If the messages are long enough the chances of not having either ‘A’ or a ‘B’ can be reduced by picking long enough messages. If m_0 and m_1 are 100 characters long, there is less that a 2% chance of getting neither an ‘A’ nor a ‘B’ in c. In general, the probabily of being able to definitely distinguish which plaintext messages of either ℓ ‘A’s or ℓ ‘B’s is approximately $1-(1-1/25)^\ell$. In the few remaining cases the $\mathcal{A}$ has to make a wild guess and so their total probability of winning is
$$1-\frac{(1-1/25)^\ell}{2}$$

Enigma fails IND-KPA & IND-COA

KPA The $\mathcal{A}$ can look at a plaintext and usually determine which ciphertext is an encryption of it by checking to see whether a letter ever encrypts to itself.

COA Statistical properties of the plaintext can be learned by counting frequency of letters in ciphertext. For example, if ‘E’ appears less frequently in c it probably appears more frequently in m.



Enigma fails IND-KPA & IND-COA

KPA The $\mathcal{A}$ can look at a plaintext and usually determine which ciphertext is an encryption of it by checking to see whether a letter ever encrypts to itself.

COA Statistical properties of the plaintext can be learned by counting frequency of letters in ciphertext. For example, if 'E' appears less frequently in c it probably appears more frequently in m.

On the whole, cryptographers don't bother modeling KPA or COA, but start with CPA as the minimum secrecy to require.

Theory and Practice

There is a big difference between being able to make good guesses about which of two plaintexts resulted in a particular ciphertext and usefully decrypting ciphertexts.

- When the various weakness were discovered, no one knew how to turn them into practical exploits.
- By 1938, various pre-computation techniques to eliminate large numbers of possibilities were developed, resulting in a few actual decryptions.
- The bomba kryptologiczna did the work (on one step in decryption) of 100 people.
- From 1939 onward, the British found new ways to exploit the weaknesses and industrialized the breaking process.

There is no such thing as a “theoretical” vulnerability.

There are, however, vulnerabilities that are yet to be exploited.

$\mathcal{A}$ is for Algorithm

$\mathcal{A}$ is for Algorithm

A scheme is *perfectly* secure (for some security notion) if there is no algorithm $\mathcal{A}$ which can win the relevant game at a better than chance rate.

Limiting the adversary

- *Perfect* security is impossible if the key is shorter than the message, as $\mathcal{A}$ could just search the entire key space and narrow down the possible plaintexts to those results
- *Asymptotic* security limits $\mathcal{A}$'s computational power and gives a negligible amount of wiggle room in the notion of winning the game at a "better than chance rate."

Reductions

Reduction: Mathematicians

- A mathematician enters a room with a sink, a table, a bucket of water on the table, a waste basket. There is a fire in the waste basket. They take the bucket from the table and douse the fire.

Reduction: Mathematicians

- A mathematician enters a room with a sink, a table, a bucket of water on the table, a waste basket. There is a fire in the waste basket. They take the bucket from the table and douse the fire.
- The next day, they enter an identical situation with the exception that the bucket is empty and next to the sink. They fill the bucket with water and place it on the table.

Reduction: Mathematicians

- A mathematician enters a room with a sink, a table, a bucket of water on the table, a waste basket. There is a fire in the waste basket. They take the bucket from the table and douse the fire.
- The next day, they enter an identical situation with the exception that the bucket is empty and next to the sink. They fill the bucket with water and place it on the table.
- “I reduced the problem to a previously solved problem.”

Proving in the right direction

Tired If you can break problem P, you can break my cipher scheme. Therefore my cipher scheme is as hard as P

Wired If you can break my cipher scheme, you can break problem P. Therefore my scheme is at least as hard as P.

Reduction: Cryptographers

“We have reduced it to a previously *unsolved* problem.”

Summary

What we've seen

- There are different security goals. (E.g., non-malleability and secrecy)
- Attack models can be defined as games (E.g., the $\mathrm{IND\text{-}CPA}$ game)
- An example of a scheme (Enigma) that tried an failed to be at least $\mathrm{IND\text{-}KPA}$ secure.
- Cryptographers try to prove that schemes meet a security notion by proving what other problems an adversary can solve if they can break the scheme.