

Die Geheimschriften
und
die Dechiffir-Kunst.

Mit besonderer Berücksichtigung
der deutschen und der französischen Sprache

von
F. W. Kasiski,
Major 3. B.



Berlin, 1863.

Druck und Verlag von F. G. Mittler und Sohn.
(Zimmerstraße 64. 65.)

CS 553

CRYPTOGRAPHY

Lecture 3

Attack models

Instructor
Dr. Dhiman Saha

- ▶ Cryptographic goals: Privacy, Integrity, Authenticity
- ▶ Historical ciphers
- ▶ Idea of Cryptanalysis

An orange circle with the text "Attack Models" inside it.

Attack Models

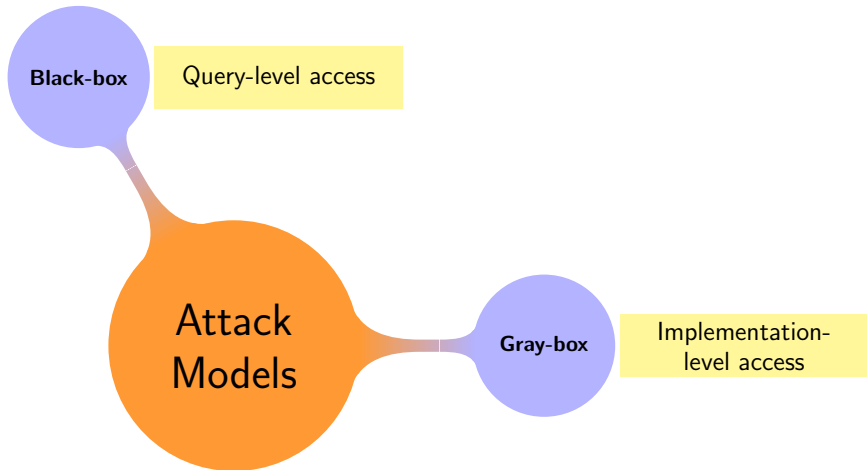
Black-box

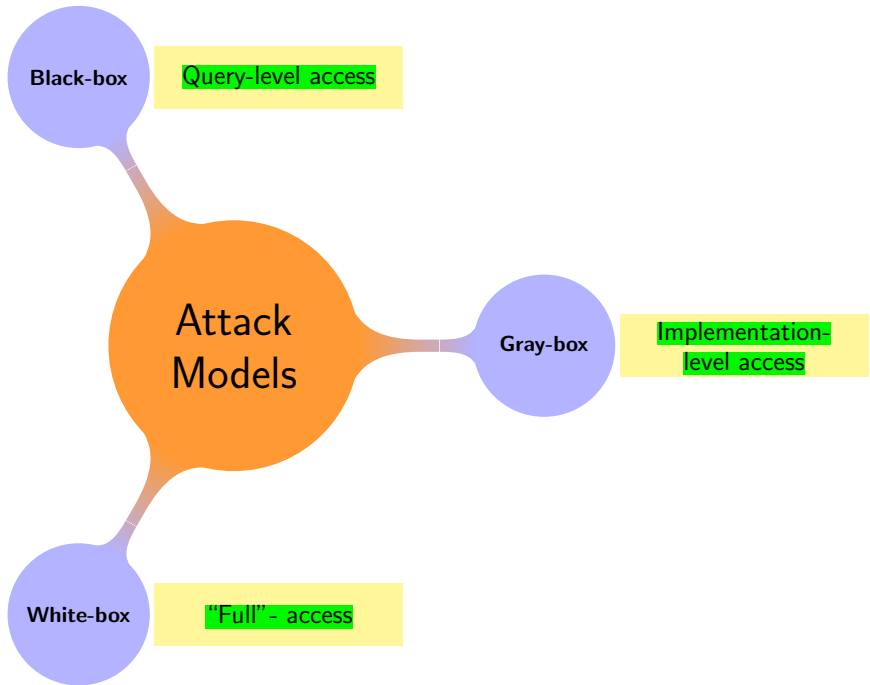
Query-level access

Attack
Models

```
graph TD; A((Attack Models)) --- B((Black-box)); A --- C[Query-level access];
```

The diagram consists of a central orange circle labeled 'Attack Models'. A purple circle labeled 'Black-box' is connected to the top-left of the orange circle by a thin, curved line. A yellow rectangle labeled 'Query-level access' is connected to the top-right of the orange circle by a thin, curved line.



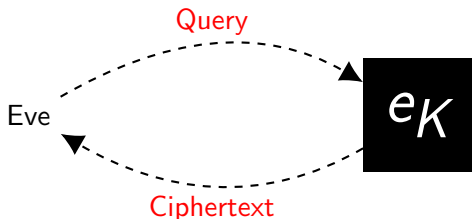



Notion of **query**

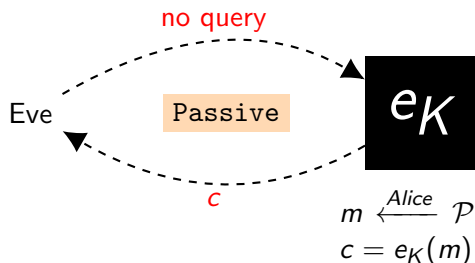
In current context


Operation that **sends an input** value to some function and gets the **output in return**, without exposing the details of that function.

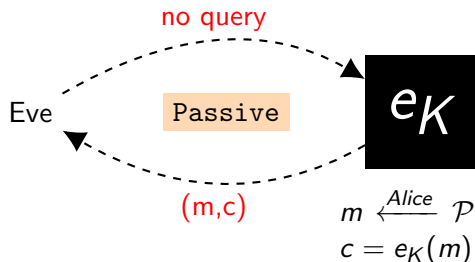
- ▶ Assume cipher behaving as a black-box
- ▶ Eve only sees what goes in and out of the cipher
- ▶ Different models based on what Eve can query for



- ▶ Eve has access to the ciphertext c only
- ▶ But **does not** know the associated plaintexts
- ▶ Or **how they were selected**
- ▶ The default scenario
- ▶ Weakest adversarial model 



- ▶ Eve can get her hands on both a ciphertext c and its corresponding plaintext m ,
- ▶ Where plaintexts are assumed to be randomly selected.
- ▶ Stronger than COA
- ▶ Is Hill Cipher KPA-vulnerable? 




A Known Plaintext Attack on the PKZIP Stream Cipher

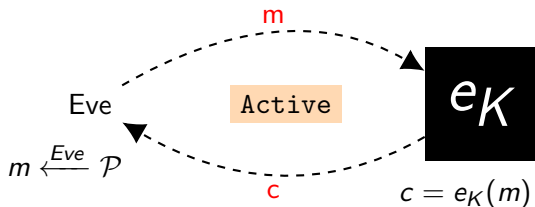
Eli Biham* Paul C. Kocher**


Abstract. The PKZIP program is one of the more widely used archive/compression programs on personal computers. It also has many compatible variants on other computers, and is used by most BBS's and ftp sites to compress their archives. PKZIP provides a stream cipher which allows users to scramble files with variable length keys (passwords).

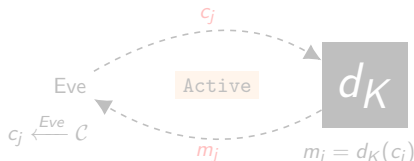
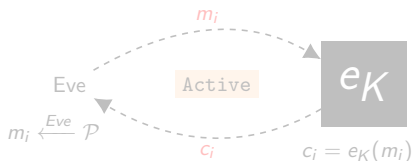
In this paper we describe a known plaintext attack on this cipher, which can find the internal representation of the key within a few hours on a personal computer using a few hundred bytes of known plaintext. In many cases, the actual user keys can also be found from the internal representation. We conclude that the PKZIP cipher is weak, and should not be used to protect valuable data.

<https://www.unix-ag.uni-kl.de/~conrad/krypto/pkcrack.html>


- ▶ Eve can perform encryption queries for plaintexts of her choice
- ▶ And observe the resulting ciphertexts.
- ▶ Notion of **active** adversary
- ▶ Stronger than KPA
- ▶ Affine Cipher is CPA-vulnerable. Why? 

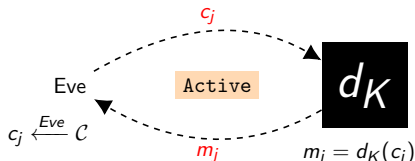
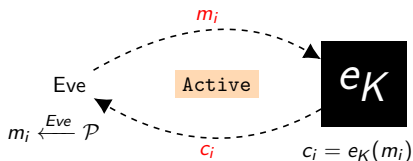


- ▶ Eve can perform encryption as well as decryption queries.
- ▶ Does this make sense? 
- ▶ Decrypting something is not always enough to break a system.
- ▶ Stronger than CPA



- ▶ More advanced: Adaptive CCA

- ▶ Eve can perform encryption as well as decryption queries.
- ▶ Does this make sense? 
- ▶ Decrypting something is not always enough to break a system.
- ▶ Stronger than CPA



- ▶ More advanced: Adaptive CCA

KPA-Security

Given one or more pairs of plaintexts and their corresponding ciphertexts, $(m_1, c_1), (m_2, c_2), \dots, (m_n, c_n)$ it must be very difficult to decrypt any ciphertext c that is **not in the given list** without knowing k .

CPA-Security

For any list of plaintexts $m_1, m_2, \dots, m_n \in \mathcal{P}$ chosen by the adversary, even with knowledge of the corresponding ciphertexts $e_K(m_1), e_K(m_2), \dots, e_K(m_n)$ it is very difficult to decrypt any ciphertext c that is **not in the given list** without knowing k .

CCA-Security

HW-Problem 

KPA-Security

Given one or more pairs of plaintexts and their corresponding ciphertexts, $(m_1, c_1), (m_2, c_2), \dots, (m_n, c_n)$ it must be very difficult to decrypt any ciphertext c that is **not in the given list** without knowing k .

CPA-Security

For any list of plaintexts $m_1, m_2, \dots, m_n \in \mathcal{P}$ chosen by the adversary, even with knowledge of the corresponding ciphertexts $e_K(m_1), e_K(m_2), \dots, e_K(m_n)$ it is very difficult to decrypt any ciphertext c that is **not in the given list** without knowing k .

CCA-Security

HW-Problem 


KPA-Security

Given one or more pairs of plaintexts and their corresponding ciphertexts, $(m_1, c_1), (m_2, c_2), \dots, (m_n, c_n)$ it must be very difficult to decrypt any ciphertext c that is **not in the given list** without knowing k .

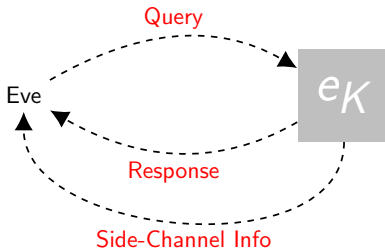
CPA-Security

For any list of plaintexts $m_1, m_2, \dots, m_n \in \mathcal{P}$ chosen by the adversary, **even with knowledge** of the corresponding ciphertexts $e_K(m_1), e_K(m_2), \dots, e_K(m_n)$ it is very difficult to decrypt any ciphertext c that is **not in the given list** without knowing k .

CCA-Security

HW-Problem 

- ▶ Eve has access to a **ciphers implementation**.
- ▶ Captures practical scenarios
- ▶ Models **physical access for applications** such as **smart cards**, embedded systems
- ▶ **Notion of Side-channel attacks**¹
- ▶ Software - timing, errors, return values
- ▶ Hardware - power, EM, faults
- ▶ Non-invasive/Invasive



¹Will be explored later in the course

- ▶ Worst-case attack model
- ▶ Eve has **full control** over the cryptographic program and its **execution environment**
- ▶ Consider ability to decrypt ciphertexts under a certain key **without sharing the key itself** - Applications like DRM
- ▶ How is that even possible?

How about embedding the key in the code itself?

Are you serious!!!

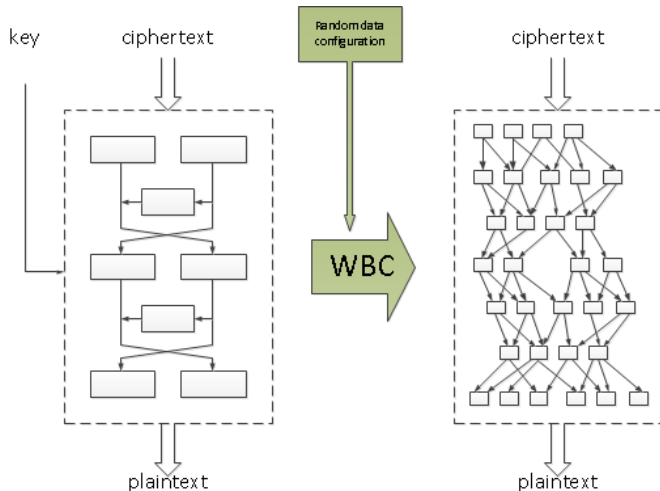
- ▶ What if this is reverse engineered?

- ▶ Worst-case attack model
- ▶ Eve has **full control** over the cryptographic program and its **execution environment**
- ▶ Consider ability to decrypt ciphertexts under a certain key **without sharing the key itself** - Applications like DRM
- ▶ How is that even possible?

How about embedding the key in the code itself?

Are you serious!!!

- ▶ What if this is reverse engineered?



- Idea: obfuscate the code to prohibit reverse engineering