Advanced Cryptography: Secure Channels

Authentication and Encryption
The Cornerstones of Cryptography (Summer 2018)

Stefan Lucks

Bauhaus-Universität Weimar

This is an **Advanced Crypto** course. Without a previous“Introduction to Cryptography”course [1], you must not take part, and your participation would, most likely, be futile. **Please take “Modern Cryptography” before taking an Advanced Crypto course like this!**

Hints:

- Ask questions! Try to **understand** what I am talking about.
- The lab session are important for you – as is solving problems from the problem sets given to you!
- I’ll switch between slides and blackboard. When I am using the blackboard, I suggest you to make notes.
- Via our web site, you will find:
  - the course slides,
  - the problem sets for the exercises / lab sessions,
  - and further information.

[1] “Kryptographie und Mediensicherheit” or “Modern Cryptography” by myself, or a similar course at the university where you previously studied.

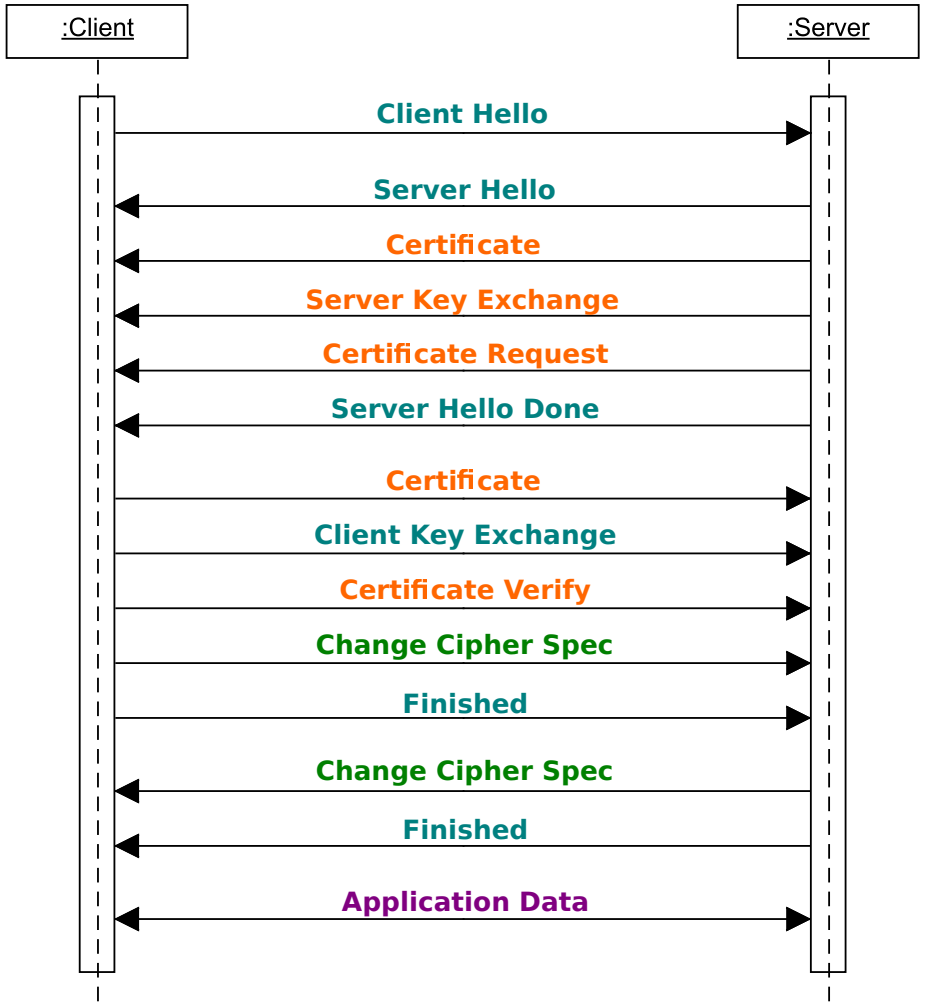
Cryptosystems at Different Levels

1. Primitives
   - Block- and stream ciphers, hash functions, trapdoor one-way functions, ...
   - AES, RC4, SHA-256, RSA, ...
2. Cornerstone algorithms and protocols
   - Authentication, encryption, key exchange, digital signatures
3. Application-oriented protocols
   - Secure session establishment, digital payment, electronic voting, ... (TLS, ...)

This lecture concentrates on the cryptographic *middleware* at the 2nd level, specifically on authentication and encryption.

TLS

. . . as an Example for an Application-Oriented Protocol

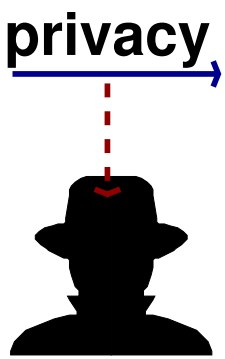


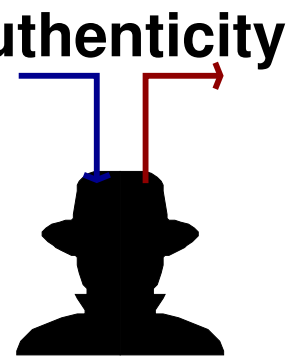
The *handshake protocol* uses **public-key encryption** and **digital signatures** to negotiate a *secret key*.

The *record protocol* employs *encryption* and *authentication* under the *secret key* to establish the **secure channel**.

Illustration: Thomas Knapke.

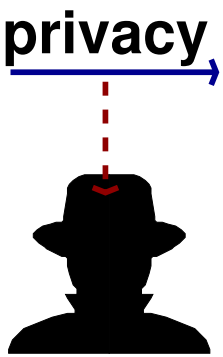
So What Are We Talking About?

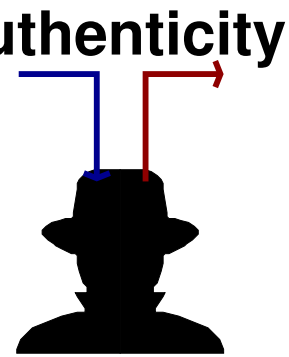




- **Message:** Data in transit from sender to receiver
- **(Data) Privacy:** Curious adversary in different models:
  - KPA (Known-Plaintext Attack)
  - KCA (Known-Ciphertext Attack)
  - CPA (Chosen-Plaintext Attack)
  - CCA (Chosen-Ciphertext Attack)
- **Encryption:** Technique to maintain (data) privacy
- **Authenticity:** Message written by the claimed sender and not modified (same as **integrity**)
- **Integrity:** Technique to detect unauthorized manipulation of a message

What Are We *Not* Talking About?

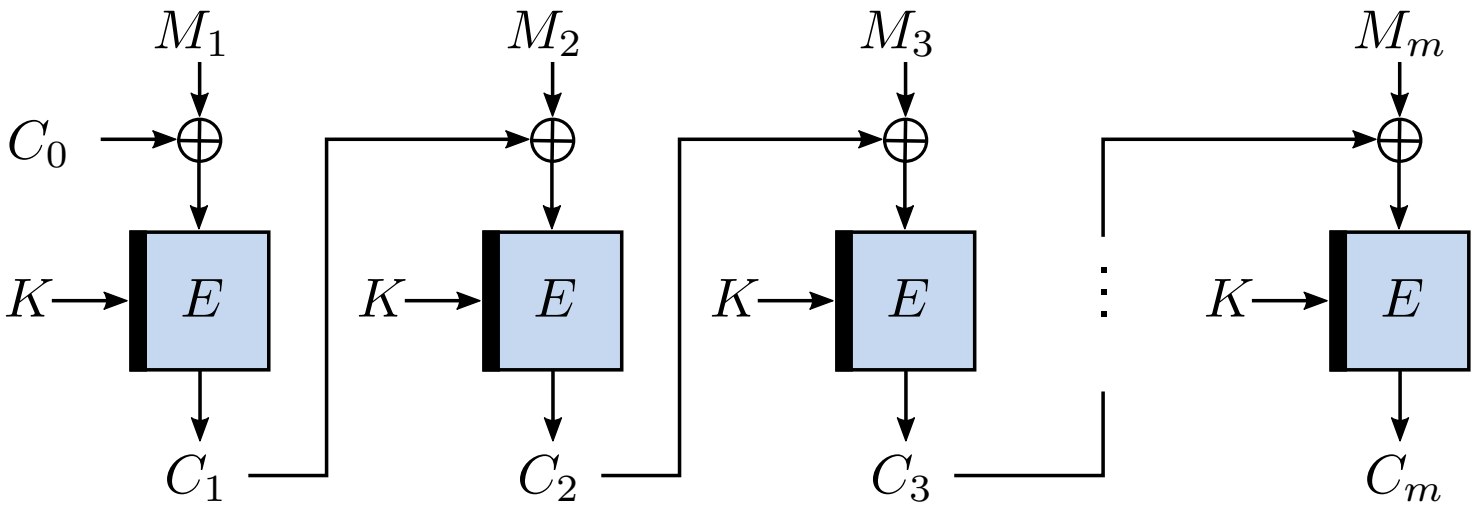




- **Traditional distinction between authenticity and integrity:**
  - *Authenticity (traditional:* Message has actually been written by the claimed sender
  - *Integrity (traditional):* Message has not been modified
  - Useful when the original message has been written, e.g., on a piece of paper
  - **Hardly useful for digital data**
- **Non-Repudiation:** The sender can prove the origin of the message to a third party (*digital signatures*)
  - More demanding than authenticity/integrity

Example: CBC Encryption

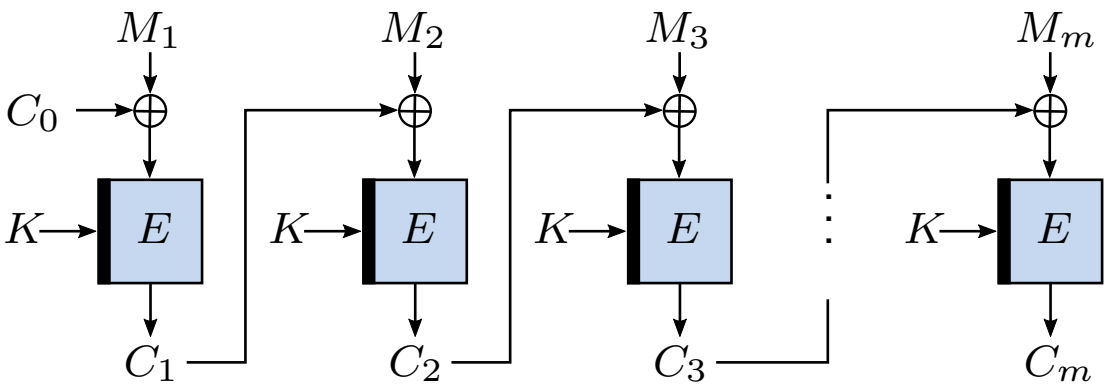
- Given a message $M_1, \ldots, M_m$
- Choose a random C_0 (the *initial value*, IV)
- Compute $C_i := E_K(M_i \oplus C_{i-1})$, for all $1 \leq i \leq m$



Why CBC Is Secure

(Some Informal Arguments)

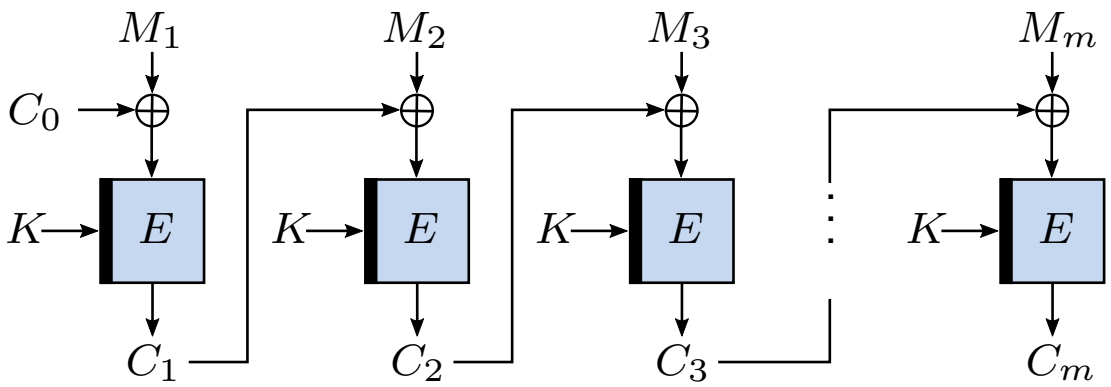
- In a *chosen-plaintext* scenario, the adversary is allowed to choose messages to be encrypted
- As long as the values $C_{i-1} \oplus M_i$ never repeat, the adversary essentially sees the encryption of independent random values (even if the same message is encrypted twice)



If the block cipher is secure, encrypted random values reveal nothing about the messages except for their lengths.

How to Break CBC Encryption (1)

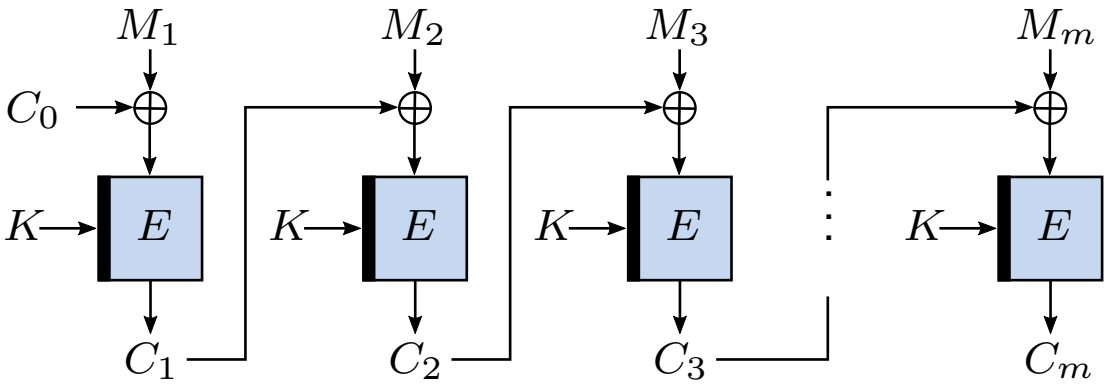
Tampering with Ciphertexts



1. $(C_i, \ldots, C_j)$ is the encryption of $(M_i, \ldots, M_j)$.
2. $(C_0 \oplus \mathbf{X}, C_1, \ldots, C_m)$ is the encryption of $(M_1 \oplus \mathbf{X}, M_2, \ldots, M_m)$ for all $\mathbf{X}$.
3. $(C_0, \ldots, C_{i-1}, C_i \oplus \mathbf{X}, C_{i+1}, \ldots, C_m)$ is the encryption of $(M_1, \ldots, M_{i-1}, M_i \oplus \mathbf{Y}, M_{i+1} \oplus \mathbf{X}, M_{i+2}, \ldots, M_m)$ for all $\mathbf{X}$ and unknown $\mathbf{Y}$.

How to Break CBC Encryption (2)

Tampering with C_0

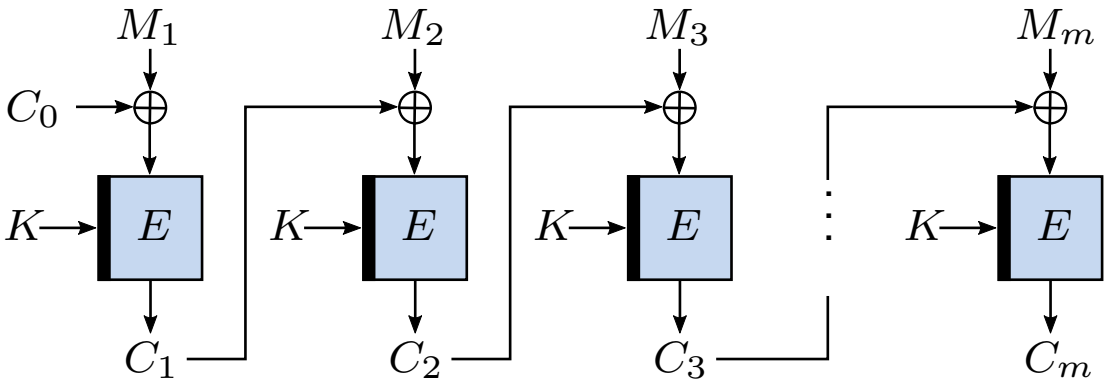


1. When using the same C_0 to encrypt several messages, it is easy to detect identical messages and common prefixes in messages
2. When the adversary can predict next C_0:
   - Given: $(C'_0, \ldots, C'_m)$ and message *candidate* $(M'_1, \ldots, M'_m)$.
   - Ask for the encryption of $(M_1, M'_2, \ldots, M_m)$ with

$$M_1 = M'_1 \oplus C'_0 \oplus C_0.$$

How to Break CBC Encryption (3)

Block-wise Adaptive Attacks



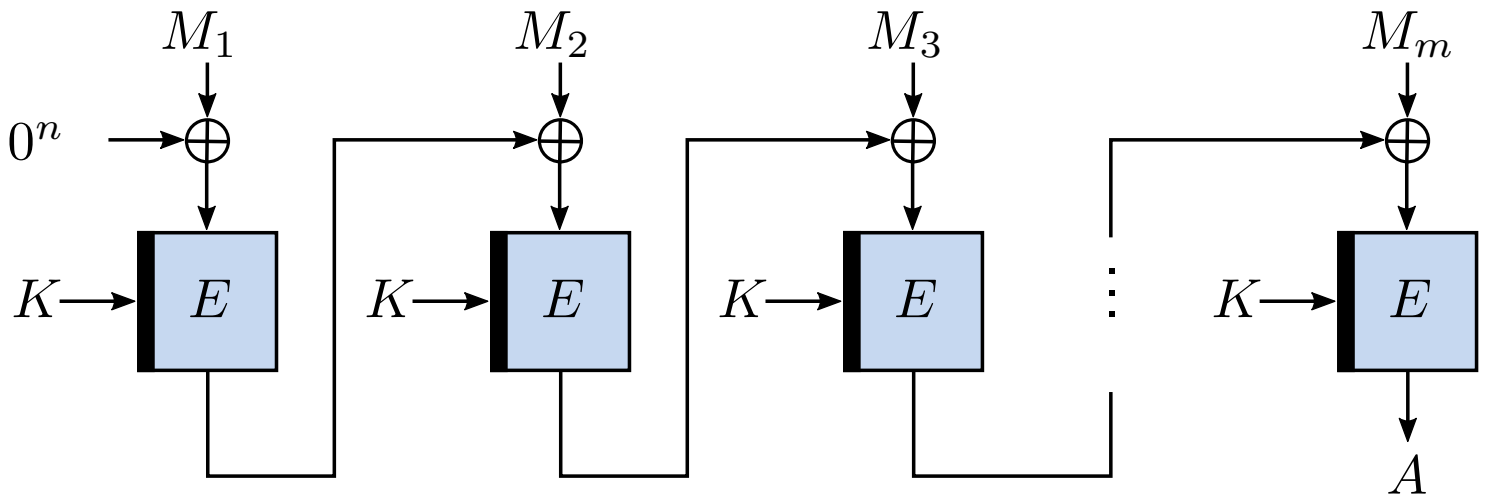
Consider a small device with restricted storage and the following interactive encryption protocol:

- Send M_1, receive C_0, C_1
- For $i \in \{1, \ldots, n\}$: send M_i, receive C_i

How would you exploit this to attack CBC?
See the problem session.

Preserving Authenticity: The CBC-MAC

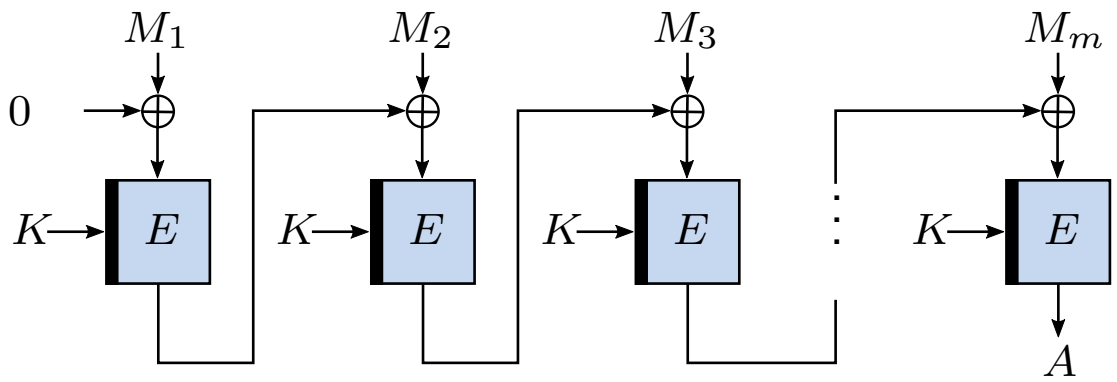
- Given message $M_1, \ldots, M_m$
- Initial value $C_0 = 0^n$
- Compute $C_i := E_K(M_i \oplus C_{i-1})$ for $1 \leq i \leq m$
- Ignore $C_1, \ldots, C_{m-1}$
- Output the "authentication tag" $A = C_m$



Why the CBC-MAC Is Secure

Some Informal Arguments

- CBC-MAC is just CBC encryption, dropping all but the final ciphertext block.
- If CBC encryption is secure, then so *should* the CBC-MAC be.



- Or does the fact that the random initial value C_0 has been replaced by a fixed 0 pose any problems?

How to Break the CBC-MAC

Forge Messages

Actually, the CBC-MAC is insecure

- If messages of different lengths are used, or
- if one uses a random C_0 instead of a fixed constant (note that the forger is allowed to *choose* C_0).

Furthermore, if one tries to establish a *secure channel* (authentic and private) by using CBC encryption and the CBC-MAC under the same secret key, this is insecure.

$\rightarrow$ See the problem session.

Are CBC Encryption and CBC-MAC secure, Or Not?

And What Is This Lecture Really About?

- **Not a meaningful question (!)** (nothing special about CBC . . .)
- Security: Some **property** to preserve under certain **attacks**
- Theory:
  - Precise **security definitions**: define the adversaries capabilities (what messages would she read or send, . . .) and her goals (what would she like to find out, tamper with, . . .)
  - Some **theorems** to discover necessary and sufficient conditions for cryptosystems meeting the security definitions
- Practice:
  - Are the capabilities of the theoretical adversary really (a superset of) those of a practical adversary?
  - Are the adversaries goals really (a superset of) what practical adversaries are interested in?