Encryption and Data Integrity Control

Holong Marisi Simalango, A.Md., S.T., M.Kom,
Gilang Bagus Ramadhan, A.Md.Kom.
Ahmadi Irmansyah Lubis, S.Kom., M.Kom.

Learning Outcome:

TP2: Students are able to **apply** practically from the concept of relevant security standards on relational databases

DATA ENCRYPTION CONCEPT

Definition of Encryption

Encryption is a fundamental solution for protecting stored data. It works by transforming readable information (**plaintext**) into an unintelligible format (**ciphertext**). This process requires credentials, like a password or passphrase, for both encryption and decryption.

Main conditions:

Encryption is a crucial method in cybersecurity that protects data in two main conditions:

1. Encrypting Data is in Transit
2. Encrypting Data is at Rest

Encrypting Data in Transit

Communication between database clients and servers, which primarily uses the **TCP/IP** protocol on standard ports (such as port 3306 for MySQL), is highly vulnerable to attacks. Hackers can easily intercept network traffic to steal sensitive queries and data because this communication is often unencrypted.

Therefore, it is crucial to implement **data-in-transit encryption**. This type of encryption protects communication by encrypting data at one point (the sender's side) and decrypting it at the other (the receiver's side).

the main point is that **encrypting database communications is crucial for securing TCP/IP sessions.**

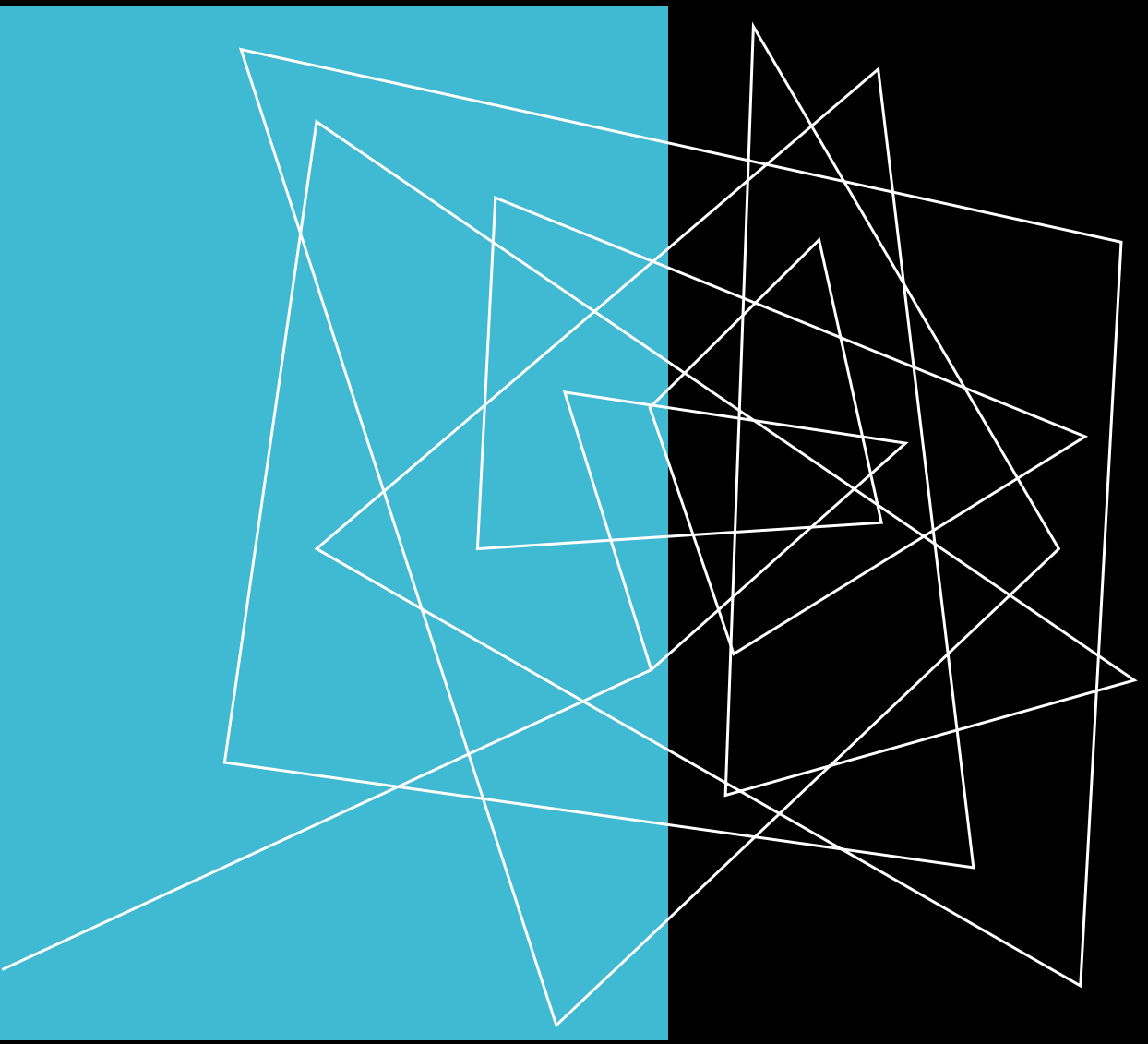
Various Encryption Options

Four main categories for encrypting database communications:

- **Database-Specific Features**: These features are built directly into the database management system (e.g., Oracle Advanced Security). This method relies on the database vendor.
- **Secure Tunnels**: Uses technologies like **Secure Shell (SSH)** to create an encrypted "tunnel" that protects data traffic.
- **Connection-Based Methods**: Uses industry standards like **Secure Sockets Layer (SSL)** to encrypt the entire communication session.
- **Relying on the Operating System**: Uses encryption features provided by the operating system, such as **IPSec encryption**.

Encrypting Data is at Rest

While Data is at Rest: Data is protected when it is inactive and stored. This is done by encrypting important files or even the entire storage device.



Data at Rest Encryption Approach: **Application-Level Encryption**

The advanced encryption standard (AES)

Encryption and decryption using the advanced encryption standard (AES), which is widely accepted as the most secure encryption approach because AES ciphertext is virtually impossible to convert back to plaintext without the proper key

The advanced encryption standard (AES) (Cont.)

The AES_ENCRYPT() and AES_DECRYPT() functions are built-in tools within a database system (specifically MySQL) designed to implement the **Advanced Encryption Standard (AES)** algorithm.

The AES standard allows for various **key lengths**, such as **128-bit, 192-bit, or 256-bit**. This key length determines the complexity of the encryption.

By default, the AES_ENCRYPT() and AES_DECRYPT() functions in MySQL use a **128-bit** key length.

The advanced encryption standard (AES) (Cont.)

There is a **trade-off** between security and performance.

- **Security:** The longer the key (e.g., 256-bit), the more difficult it is for unauthorized parties to break the encryption.
- **Performance:** The longer the key, the more computational resources are required, making the encryption and decryption processes slower.

The advanced encryption standard (AES) (Cont.)

AES_ENCRYPT() encrypts the string str using the key string key_str, and returns a binary string containing the encrypted output.

AES_DECRYPT() decrypts the encrypted string crypt_str using the key string key_str, and returns the original (binary) string in hexadecimal format.

Types of encryption supported by MySQL

Two primary types of encryption supported by popular database management systems (DBMS) like MySQL:

- **Symmetric Key Encryption:** Uses the same key for both encrypting and decrypting the data.
- **Asymmetric Key Encryption:** Employs a pair of different keys—one for encryption and the other for decryption.

Key_str

key_str is the encryption key argument. It can be the encryption key itself or the base material for creating a key using a **Key Derivation Function (KDF)**. The most important point is: for the same instance of data, the key_str value **must be exactly the same** for both encryption (AES_ENCRYPT()) and decryption (AES_DECRYPT()).

Encryption and Compression Functions:

Table 14.18 Encryption Functions

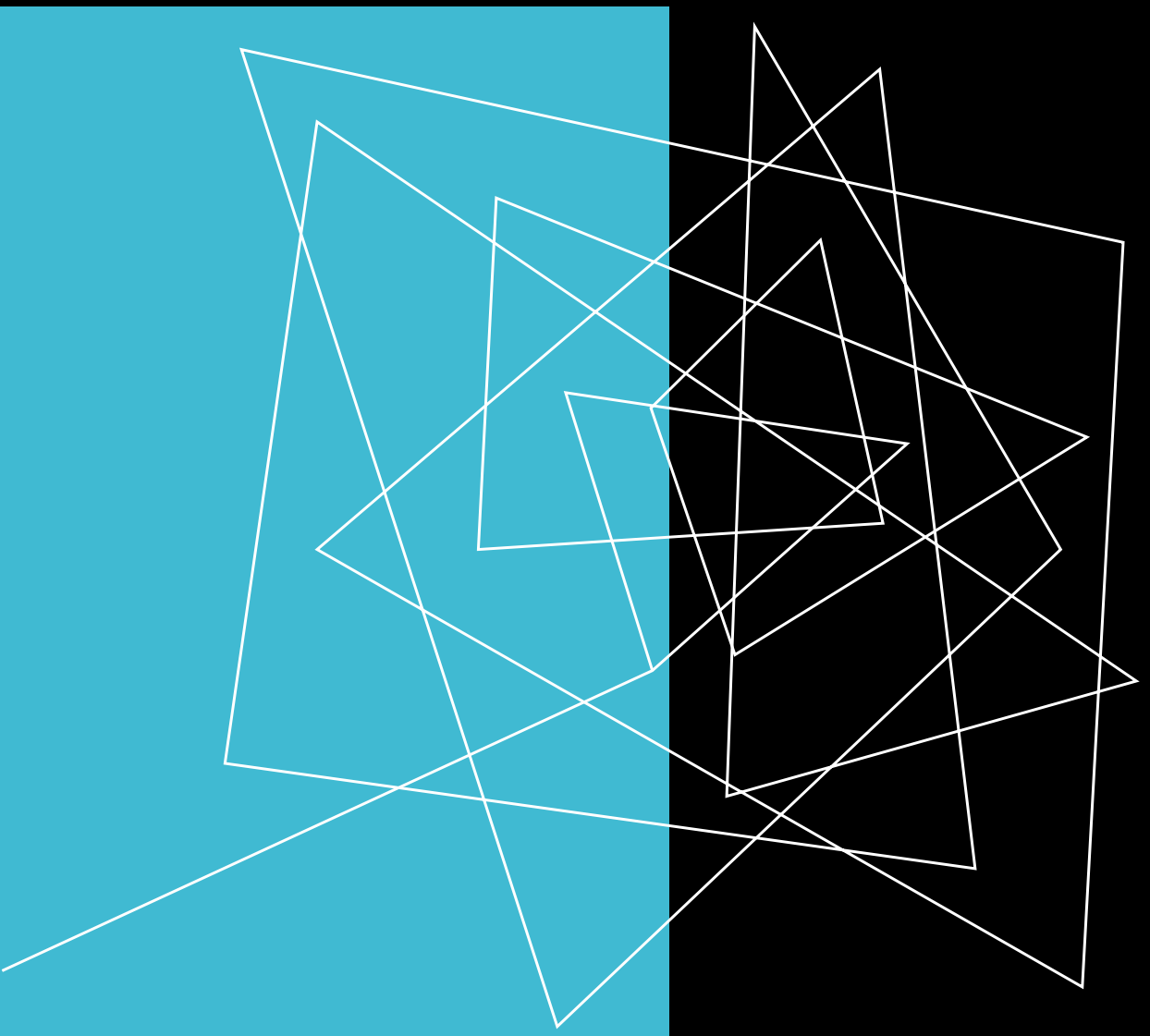
Name	Description
AES_DECRYPT()	Decrypt using AES
AES_ENCRYPT()	Encrypt using AES
COMPRESS()	Return result as a binary string
MD5()	Calculate MD5 checksum
RANDOM_BYTES()	Return a random byte vector
SHA1(), SHA()	Calculate an SHA-1 160-bit checksum
SHA2()	Calculate an SHA-2 checksum
STATEMENT_DIGEST()	Compute statement digest hash value
STATEMENT_DIGEST_TEXT()	Compute normalized statement digest
UNCOMPRESS()	Uncompress a string compressed
UNCOMPRESSED_LENGTH()	Return the length of a string before compression
VALIDATE_PASSWORD_STRENGTH()	Determine strength of password

Encrypting Data in Transit

While Data is at Rest: Data is protected when it is inactive and stored. This is done by encrypting important files or even the entire storage device.

One may have acquired significant information regarding the database server functioning as a networked service. It has been discovered that the majority of database settings utilize TCP/IP as their communication protocol.

MySQL uses port 3306.



SESI PRAKTIKUM

AN APPLICATION STORES MD5() STRING VALUES IN A CHAR(32) COLUMN (PRAKTIKUM)

```
CREATE TABLE md5_tbl (md5_val
CHAR(32), ...);

INSERT INTO md5_tbl (md5_val, ...)
VALUES(MD5('abcdef'), ...);
```

MODIFY THE APPLICATION TO USE UNHEX() AND BINARY(16) (PRAKTIKUM)

```
CREATE TABLE md5_tbl (md5_val
BINARY(16), ...);

INSERT INTO md5_tbl (md5_val, ...)
VALUES(UNHEX(MD5('abcdef')), ...);
```

HEX()
(PRAKTIKUM)

SELECT UNHEX('4D7953514C');

Hasilnya : MySQL

SELECT X'4D7953514C';

Hasilnya juga: MySQL

SELECT UNHEX(HEX('string'));

Hasilnya: 'string'

Bagaimana dengan NULL?

AES_ENCRYPT (PRAKTIKUM)

```
INSERT INTO tbl_A

VALUES
(1,AES_ENCRYPT('text',UNHEX('F3229A0B371ED2D9441B830D21A
390C3')));
```

```
INSERT INTO tbl_A

VALUES

(1,AES_ENCRYPT('text', UNHEX(SHA2('My secret
passphrase',512))));
```

THANK YOU