AnYong Database Encryption System User Guide

(Intel SGX VM image releases)

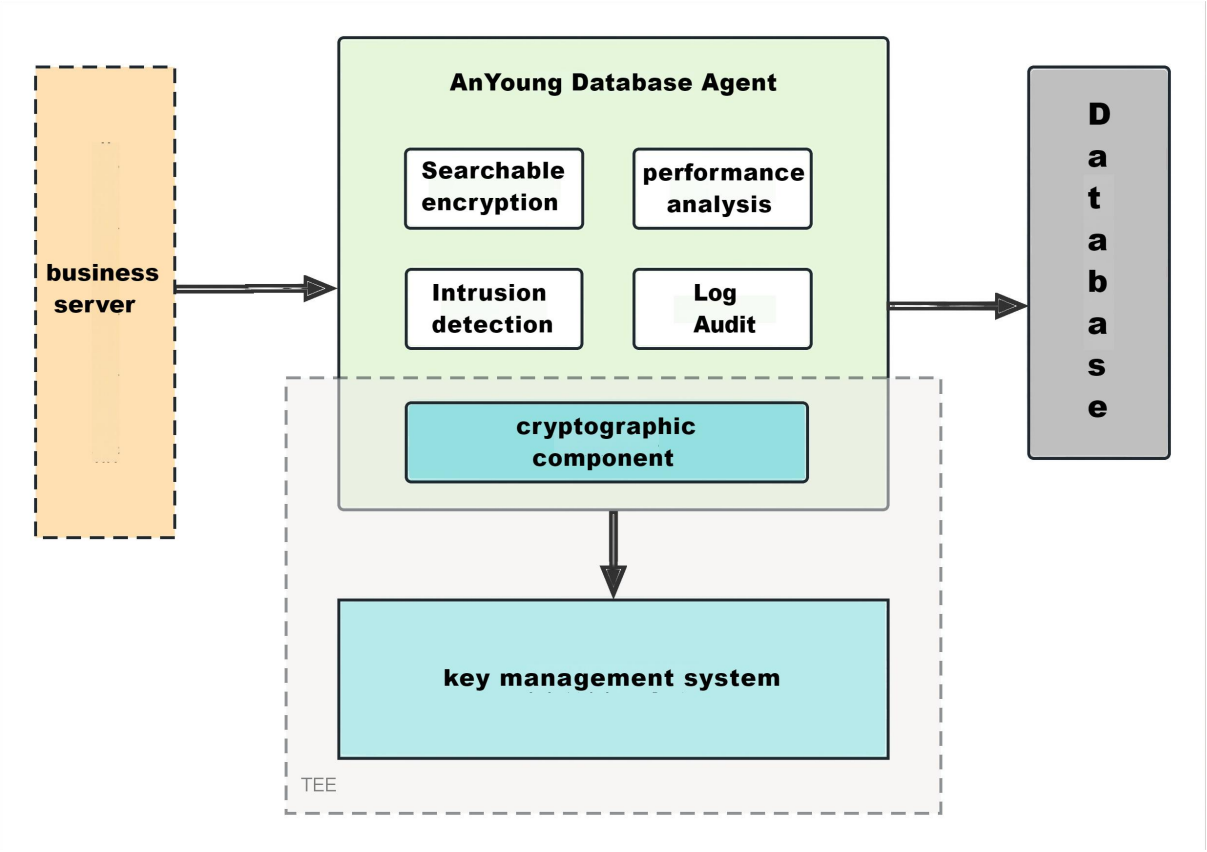
Hefei Anyong Information Technology Co.

Index

AnYoung Database Encryption System Instruction Manual

一、Products

AnYoung Database Encryption System is a database leakage prevention product based on Searchable Encryption, Trusted Execution Environment and Transparent Encryption to realize encryption and storage of sensitive data. The system supports the use of encryption algorithm SM4 to encrypt sensitive data, supports different fine-grained encryption configurations such as columns、 tables and libraries, and can be applied to encryption of structured data in relational databases. It can effectively satisfy the needs of various types of users in terms of equal protection, sub-protection assessment, and data security protection.



1. Product Advantages

（1）Based on TEE technology

Key management and usage are restricted to within the TEE, and keys are not explicitly stored in memory to ensure key security.

（2）Chip-level key security

Based on chip-level key protection technology to ensure key security.

（3）Efficient encrypted index

Unique and engineering-proven searchable encryption technology ensures efficient indexing after encryption and solves industry pain points.

（4）Independent key management

Support software, cryptograph, encryption card and other key generation methods, encryption key independent generation, independent management.

2. Functional Features

（1）Dynamic encryption and decryption

Encryption and decryption to intelligent data stored in the database for automatic real-time flexible dynamic encryption and decryption, without human intervention, to fully realize the safe storage and safe use of data, to prevent the information involved in the confidentiality.

（2）High Disaster Tolerance

Supports dual-computer deployment of encryption system with master-slave mutual backup to prevent business loss caused by host failure, network failure and program failure. A series of reliability protection technologies, such as batch encryption and decryption, data verification protection, database operation status monitoring, pre-encryption verification, backup recovery, etc., ensure the security, reliability and recoverability of data encryption and decryption process.

（3）Fully transparent access

After encryption, it maintains full transparency to the outside world and supports operations such as query, insertion, update and deletion, and the application system does not need to be modified. No modification of client statement operation and application program is required; it is fully transparent to the application development interface; common third-party management tools can be used normally; and it is transparent to stored procedures and functions.

（4）Independent Authority and Control

Provide privilege control for encrypted data, restricting the addition, deletion, modification and checking operations of some users from the client's program, IP, time and day of the week in multiple dimensions to prevent the leakage of sensitive data. Provide automatic encryption/decryption and transparent access to authorized users without any perception, and provide third-party independent access control to unauthorized users independently of the database authority system.

（5）Support for multiple data types and platforms

It can support various database types such as ***Oracle***, ***MySQL***, ***PostgrelSQL***, etc. and Linux operating system.

3．Applied value

（1）Hacker-proof database dragging

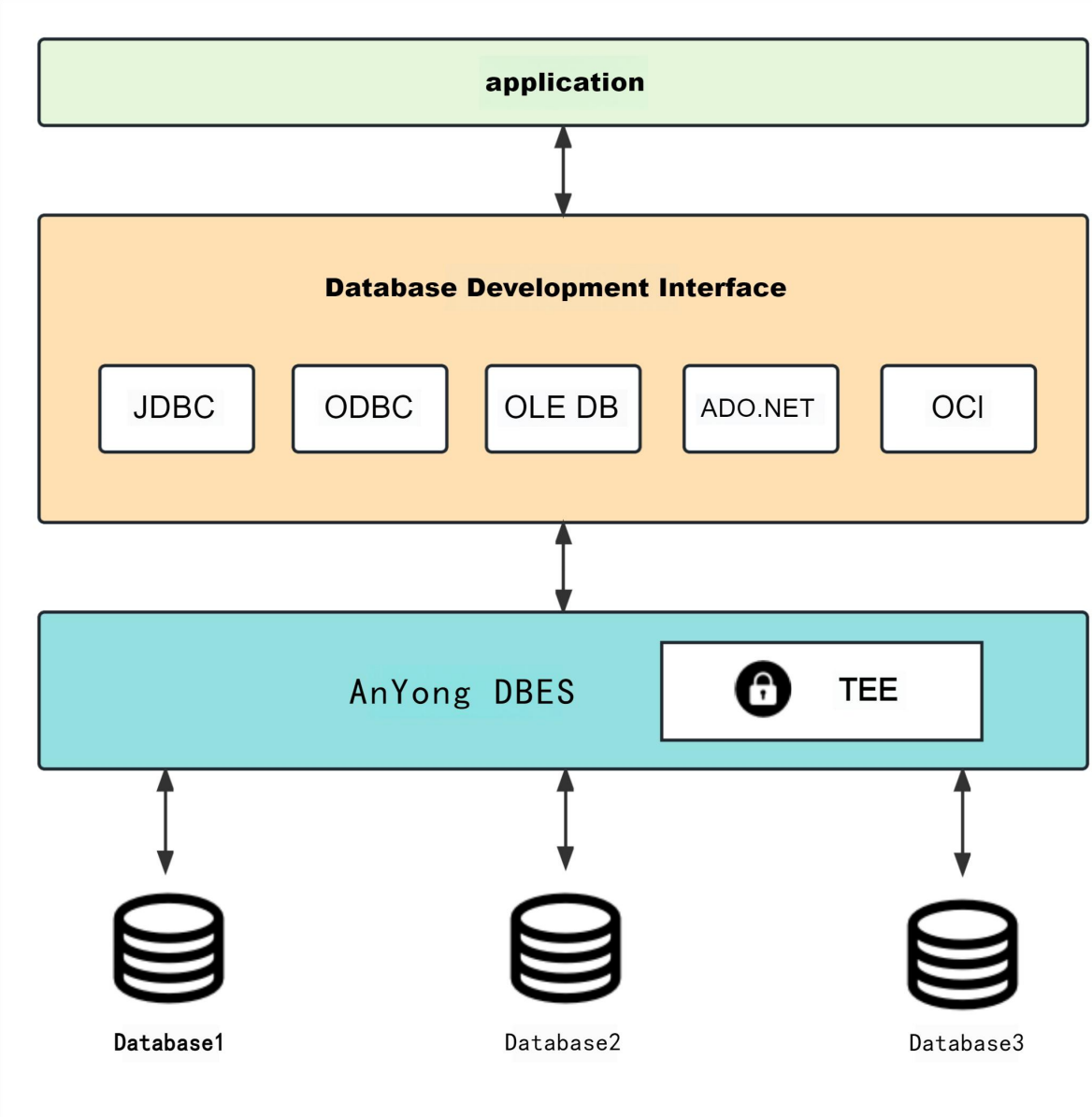
Through the database transparent encryption products to encrypt sensitive data storage, to ensure that others even if they get the data files, but also can not access sensitive information. It can effectively prevent hackers from utilizing database loopholes to attack and invade the database, and ultimately drag the database against the data in the database.

（2）Prevention of unauthorized access

Database maintenance personnel, who generally have DBA privileges, have access to all sensitive information in the database, which poses the risk of insider leakage and does not comply with security norms. Through independent and enhanced privilege control and separation of powers mechanism, it ensures that no user can access sensitive information in the database without obtaining ciphertext privileges.

4. Deployment model

The data encryption product adopts bypass deployment mode, which can be routed to the customer's database system without changing the existing system and network topology. After the external device is deployed, it will be automatically deployed in the customer's database system after the system function is turned on, without manual intervention, simple and fast.



二、SGX Version Introduction

Through this platform, we hope to learn and communicate with users and explore the future development of trusted execution environment. We welcome users to provide valuable suggestions and feedback at any time, and your participation is crucial to us. We will always listen to every user's voice and continue to optimize and improve the SGX version to better serve the needs of users.

1. Introduction to Encryption Modules

（1）Usage and Parameters

The project is divided into two modules, the key management program as well as the database encryption agent program.

• ***crypto_gm*** is the key management program of Ernst & Young database encryption system, which is responsible for the initialization and backup of the system key.

PATH：*./bin/crypto_gm*

Available options include:

parameters	functionality
-i	Generating Keys
-h, --help	help
--sym	Symmetric (import/export) keys
--asym	Asymmetric (import/export) keys
-d, --import	Import Keys
-e, --export	Exporting Keys
-p, --pin	User-configured PIN value
--client_id	Configure the client id
--file	Key File
--password	AES128 keys，HexString
--pubkey	SECP256R1 public key，HexString

The following is sample code for generating a key, importing and exporting a symmetric key, and importing and exporting an asymmetric key:

Initialize system key:

```
// -i initialize key --pin user-configured PIN value --client_id configure client_id
./crypto_gm -i --pin 12345678 --client_id testid
```

Derive keys symmetrically:

```
// -e Export key --sym Symmetric encryption, using AES128 algorithm --pin User-configured PIN value --client_id Configure client id --password AES128 key, in HexString form --file Key file
./crypto_gm -e --sym --pin=12345678 --client_id=testid \
--password=f79727ef29d0fe792c196c4bafcb3fb9 --file key.sym
```

Import keys symmetrically:

```
// -d,--import import key --sym Symmetric encryption, using AES128 algorithm --pin User-configured PIN value --client_id Configure client id --password AES128 key, in HexString form --file Key file
./crypto_gm -d --sym --pin=12345678 --client_id=testid \
--password=f79727ef29d0fe792c196c4bafcb3fb9 --file key.sym
```

Derive keys asymmetrically:

```
// -e Export key --asym Asymmetric encryption, using SM2 algorithm --pin User-configured PIN value --client_id Configure client id --pubkey SECP256R1 public key, in HexString form -file Key file
./crypto_gm -e --asym --pin=12345678 --client_id=testid1 \
--pubkey=42A48FEB55BBFF2F8E310B2D77D98D7125B350A97A4AFC95EBCC4E64F7B\
99D5A030C7E966819B99D72F9DA1FBAA9C27B2E89ACD846B5FACC33CC3ACDADDA4DD7 \
--file keya.sym
```

Importing keys asymmetrically:

```
// -d,--import import key --asym asymmetric encryption, using SM2 algorithm --pin user-configured PIN value --client_id configure client id --file key file
./crypto_gm -e --sym --pin=12345678 --client_id=testid \
--password=f79727ef29d0fe792c196c4bafcb3fb9 --file key.sym
```

• ***db_shield-server*** The main executable file for the EY Database Encryption System. To ensure that the system starts successfully according to your needs, we provide the following parameters for you to select and configure.。

PATH：*./bin/db_shield-server*

Available options include:

parameters	**functionality**
--postgresql_enable	Enable this parameter to activate the proxy function to perform encryption and decryption operations on the postgresql database.
--mysql_enable	Enable this parameter to activate the proxy function to perform encryption and decryption operations on the MySQL database.
--tee_enable	Mandatory parameter for the SGX Trusted Execution Environment. Selecting this parameter indicates that you want to start the program in the SGX Trusted Execution Environment.
--db_port	Specifies the communication port number for the database.
--db_host	Specify the IP address of your database.
--encryptor_config_file	Provides the path to the configuration file that describes the specifics of the encryption and decryption operations and the target database tables.
--client_id	unique identification number assigned to the client.
--incoming_connection_host	Set the startup IP address of the encryption agent. By default, this address is 0.0.0.0.
--incoming_connection_port	Specifies the communication port number of the encryption proxy server. By default, this port is 9393.

After initializing the key with ***crypto_gm***, you can start the AnYong database encryption system with the following code：

```
./db_shield-server --mysql_enable --tee_enable --db_port=3306 \
--db_host=127.0.0.1 --encryptor_config_file=search_test.yml \
--client_id="testid"
```

三、Quick Start

1. Quick Start Installation Scripts

```
1.Run . /install.sh
2.Read ReadMe.txt
3.(Optional) Adjust as needed . /crypro_gm.sh parameters in the script.
4.(Optional) Adjust as needed . /server.sh parameters in the script.
5.(Optional) Write database encryption configuration file on demand,
e.g. . /bin/test_search.yml
6. Run . /crypro_gm.sh to generate the key.
7.Run . /server.sh to start the service with the default configuration.
```

2. Test program

（1）Preparation

This document uses the operation of PHP files on a MySQL database as a test example. Before testing, please make sure that PHP and its related dependencies are installed:

```
//For MySQL:
sudo apt install php php-mysql

//For PostgreSQL:
sudo apt install php php-pgsql
```

（2）Design and create encryption tables

To design a table that needs to be encrypted, the columns that need to be encrypted must be defined as blob types.

For example, in table test, the name_encrypted and age_encrypted columns need to be encrypted, so the table test needs to be created with these two columns defined as blob types.

The ayx_name_encrypted_searchindex field also needs to be configured to represent that the name_encrypted field can be searchably encrypted and is also of type blob.

列名	是否加密	类型
name_encrypted	是	blob
name_unencrypted	是	blob
ayx_name_encrypted_searchindex	是	blob

```
CREATE SCHEMA IF NOT EXISTS test;
```

```
CREATE TABLE IF NOT EXISTS test.test (
`id` int(11) NOT NULL AUTO_INCREMENT,
`name_encrypted` blob DEFAULT NULL,
`name_unencrypted` blob DEFAULT NULL,
`age_encrypted` blob DEFAULT NULL,
`age_unencrypted` blob DEFAULT NULL,
`ayx_name_encrypted_searchindex` blob,
PRIMARY KEY (`id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

（3）Encryption proxy service configuration(default configuration file provided)

Configure the encryptor_config_file parameter during startup, specifying the search_test.yml file. The contents of the file are as follows:

```
#default parameter
defaults:
  #Specify encryption and decryption methods
  crypto_envelope: dbshieldblock
  #Default searchable encrypted substring length, 5 means the smallest searchable encrypted substring length is 5.
  substring_length: 5
#Databases requiring encryption
schemas:
  #Table to be encrypted
  - table: test
    #Column names
    columns:
      - id
      - name_encrypted
      - name_unencrypted
      - age_encrypted
      - age_unencrypted
    #Column info to be encrypted
    encrypted:
      #Column name
      - column: name_encrypted
        #Type of return result str for string int32 for int 32-bit type
        data_type: "str"
        #Whether searchable encryption operations can be performed, true means yes, no configuration defaults to false, true corresponds to ayx_column_searchindex field will have the value
        searchable: true
        #The minimum searchable substring length of the column can be configured individually, and defaults to the value configured by the default parameter if it is not set.
        substring_length: 2
      - column: age_encrypted
        data_type: "int32"
```

（4）PHP file preparation (default test file provided)

```php
<?php
function test($server, $port)
{
   echo "$server:$port\n";
   $conn = @mysqli_connect($server, 'root', '123456', 'test', $port);
   while (!$conn){
     sleep(1);
     $conn = @mysqli_connect($server,'root','123456','test',$port);
  }
  $conn->query('CREATE SCHEMA IF NOT EXISTS test;');
  $conn->query("CREATE TABLE IF NOT EXISTS test.test (
                  id int(11) NOT NULL AUTO_INCREMENT,
                  name_encrypted blob DEFAULT NULL,
                  name_unencrypted blob DEFAULT NULL,
                  age_encrypted blob DEFAULT NULL,
                  age_unencrypted blob DEFAULT NULL,
                  ayx_name_encrypted_searchindex blob,
                  PRIMARY KEY(id))ENGINE=InnoDB DEFAULT
                  CHARSET=utf8mb4;");
   if($port==3306){
   echo "Directly using the database query results in:\n";
  $sql="select convert(name_encrypted using utf8mb4),convert(name_unencrypted
using utf8mb4),convert(age_encrypted using utf8mb4),convert(age_unencrypted
using utf8mb4) from test.test limit 500;";
   }else{
   echo "Connect to the encryption agent and insert data into the
table:('test','test','12','12')\n";
   $conn->query("INSERT INTO
test.test(name_encrypted,name_unencrypted,age_encrypted,age_unencrypted)
VALUES('test','test','12','12')");
   echo "The result of the query using an encrypted proxy is:\n";
   $sql="select * from test.test limit 500 ";
 }
 $res=$conn->query($sql);
 if($res) {
  $rows=$res->fetch_all();
  print_r($rows);
 }else{
  echo mysqli_error($conn).':'.mysqli_error($conn);
 }
 mysqli_free_result($res);
 $conn->close();
  echo PHP_EOL;
}
test('127.0.0.1','9393');
test('127.0.0.1','3306');
```

（5）Execution of PHP files

```
php test.php
```

（6）Observing the results

The original text can be visualized through the Go agent, while only the unencrypted fields can be viewed through the MySQL database.

（7）Database Query Validation

```
mysql -uroot -p
use test;
select * from test \G;
```

Perform a fuzzy search

```
select * from test where name_encrypted like 'te%';
```

As seen from the query results, the same data changes after encryption, but the data remains the same after decryption, realizing transparent encryption. And Go agent is able to realize fuzzy search, while native database is not.

四、Frequently Asked Questions (FAQ)

1. -bash: Permission denied

```
// Add executable permissions to start_server.sh and crypto_gm.
sudo chmod u+x start_server.sh
sudo chmod u+x start_crypto_gm.sh
sudo chmod u+rwx ./bin/enclave/enclave.signed
sudo chmod u+x ./bin/getlicense
sudo chmod u+x ./bin/db_shield-server
sudo chmod u+x ./bin/crypto_gm
```

2. FATA[0000] Please Make Sure Start In Azure with subscribed image !<nil>

Please start this service in your Azure Marketplace subscription's virtual machine image product.

3. Failed to open Intel SGX device.

Please use the recommended configuration or make sure your sgx driver is enabled.

Run ls /dev/sgx* to ensure that the driver can be retrieved.

五、Contact Us

Company: Hefei Anyong Information Technology Co.

Address: Room 1101, Block A, Building J1, No. 2800, Innovation Avenue, Hi-Tech Zone, Hefei City, Anhui Province, China

Website: www.anyong.net

Email： support@anyong.net

Tel： +86 18055100335