SYBASE®

Sybase® Adaptive Server® Enterprise Encryption Option

Protecting Sensitive Data

PRODUCT DATASHEET

Sybase's Encryption Option for Adaptive Server Enterprise allows you to protect sensitive data on disk while minimizing the performance impact and without exposing encryption keys.

CLOSING THE SECURITY GAP

Critical personal data from thousands of clients has been exposed due to lost computers, backup tapes and disk drives—and when identity thieves have broken into systems. If the data is stored in readable form, thieves can bypass database access controls with utilities that read the physical blocks of data and recover sensitive information.

To keep your data out of the wrong hands, the ASE Encryption Option can be added to Adaptive Server Enterprise. It will allow you to selectively encrypt sensitive data, ensuring its protection even if the media containing the data is lost or stolen.

ON-DISK ENCRYPTION THAT MEETS THREE CRITICAL CRITERIA

Sybase's ASE Encryption Option offers a number of distinct advantages that allow on-disk encryption to be more easily managed and still provide a much higher level of protection. Most importantly:

- **Performance Impact Minimized.** Encrypting data with the Sybase ASE Encryption Option is done at the column-level. This makes it easy to encrypt personal data such as customers' social security numbers without encrypting less sensitive data such as the state they live in. Because performance is easier to maintain when we minimize the number of encryptions/decryptions the database needs to do, this column-level encryption is much more efficient. With the native encryption provided by Sybase ASE Encryption Option, table owners can quickly add encryption to existing tables. For example, to encrypt the ssn column of the *employee* table with a key named *ssn_key*, you would issue the command:

  ```
  alter table employee modify ssn
  encrypt with ssn_key
  ```

- **Encryption Keys Not Exposed.** To avoid the pitfalls of external keys which are transmitted over a network, the Sybase ASE Encryption Option manages the keys within the server and keeps the keys encrypted. Keys are stored in the database's *sysencryptkeys* table. For example, to create a key called *ssn_key* for the AES encryption algorithm, the command is:

  ```
  create encryption key ssn_key for AES
  ```

  After this command is issued, a new key, *ssn_key*, is ready for use.

- **No Application or Schema Modifications Required.** Sybase ASE Encryption Option does not require application modifications. Instead it allows your existing databases with existing applications to encrypt data through database security administration. Because table schemas remain intact, your queries and data manipulation code need not be touched.

FLEXIBLE, EASY-TO-USE ENCRYPTION

Using a simple, direct, scriptable syntax, the Sybase ASE Encryption Option allows you to select different encryption methods for different types of data.

Permission-based System Controls

For better protection from both internal and external data breaches, users and groups are given permission to decrypt the data, as opposed to requiring users and applications to communicate encryption keys. Users without permission are unable to see cleartext data. Table owners easily manage permission controls through extensions to the GRANT and REVOKE syntax. For example, to give users with the role account_manager_role decrypt permission on the ssn column of the employee table, the table owner uses the command:

```
grant decrypt on employee(ssn) to account_manager_role
```

Here only users with the account_manager_role would see the decrypted ssn column, while all others would get a permissions error. With this permission-based system, even a system administrator could be denied access to view certain data.

Advanced Protections

For data which takes on very few values, additional protection is available. For example, in a voting system which records "Yes" or "No" votes, if we had no additional protection, all "Yes" votes might be encrypted as "0x4609c2fa" and all "No" votes as "0xa123b4e1". While this might make it easy to create joins on this table on the vote column, it also means that an unauthorized person need figure out only how one vote was recorded to discover all the rest.

The solution for this kind of problem is an initialization vector of pseudo-random data used to mask the data values before encrypting them. With ASE native encryption, you can do this when creating the encryption key. For example, to create a key called vote_key that has a random initialization vector, the command would be:

```
create encryption key vote_key for AES
with init_vector random
```

Changing keys is also easy with the Sybase ASE Encryption Option. It's a simple two-step process: create a new key, encrypt with the new key. For example:

```
create encryption key new_ssn_key for AES
alter table employee modify ssn
encrypt with new_ssn_key
```

Integration with ASE utilities. The Sybase ASE Encryption Option will become an integrated part of your server. Key utilities such as bulk copy (bcp), schema generation (DDLGen), the migration tool and Sybase Central have been updated to support the ASE Encryption Option. Sybase has also upgraded key technologies such as Sybase Replication Server® to support encrypted data.

ADAPTIVE SERVER ENTERPRISE DATA SECURITY AND DIRECTORY SERVICES OPTION

Complimentary to the Sybase ASE Encryption Option, Sybase offers ASE Security and Directory Services Option ensuring data privacy through row-based access controls, the encryption of in-transit data, and support for LDAP, Active Directory, and PAM services. Visit **www.sybase.com/ase** to learn more about the Security and Directory Services Option. Together, ASE's security options provide an extremely high level of data protection.

Sybase, Inc.
Worldwide Headquarters
One Sybase Drive
Dublin, CA 94568-7902
U.S.A
1 800 8 sybase

www.sybase.com

SYBASE®