Transparent Data Encryption

Douglas R. Vanek
Senior Systems Engineer
Monday, October 7, 2013

PROGRESS EXCHANGE 2013
DISCOVER. DEVELOP. DELIVER.

CNET News

Apple loses another unreleased iPhone (exclusive)

An Apple employee lost yet another unreleased iPhone in a San Francisco bar last month, leading to an investigation by San Francisco police and Apple security, CNET has learned.

by Greg Sandoval and **Declan McCullagh** | August 31, 2011 12:45 PM PDT

Cava22, the San Francisco bar where another unreleased iPhone apparently went missing.

Remember this?

Source: http://news.cnet.com/8301-13579_3-20099899-37/apple-loses-another-unreleased-iphone-exclusive/

First, a Note about SECURITY

Security is not a solution, but a process…

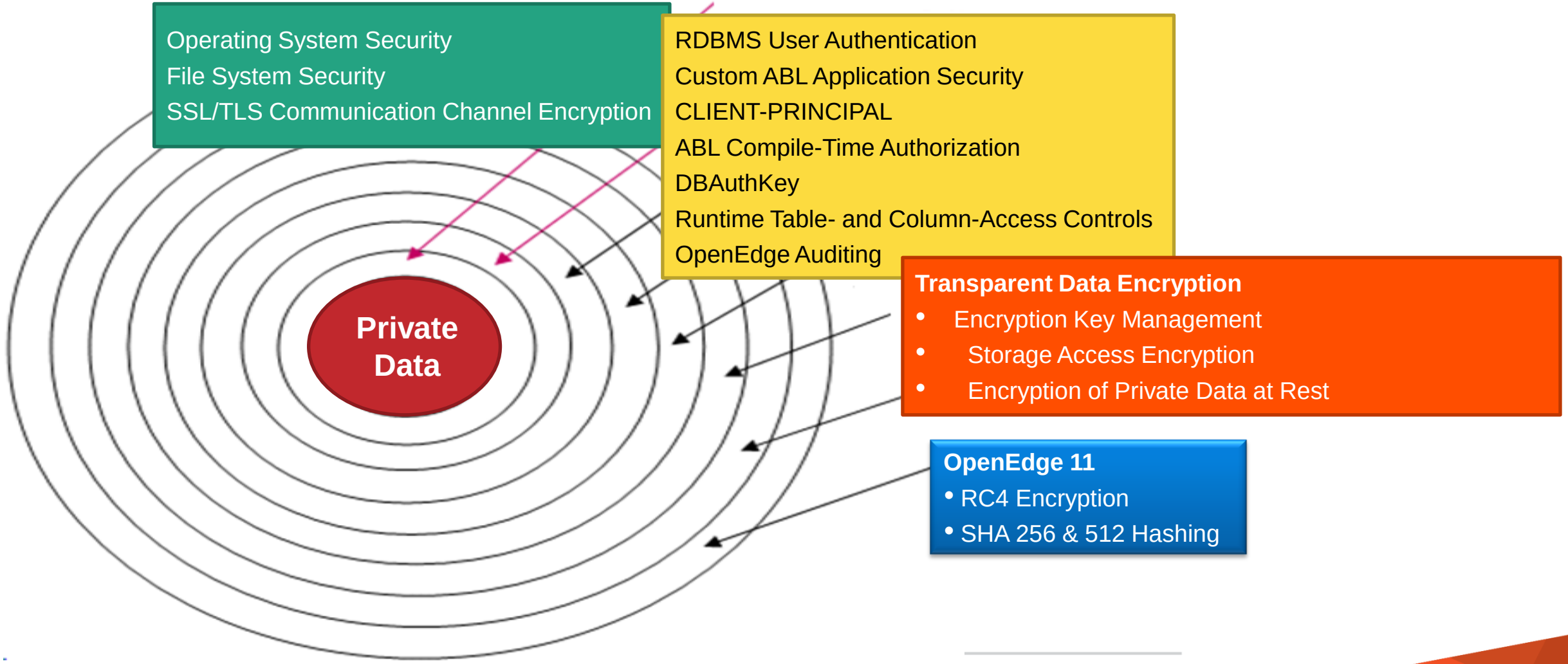
…which needs a set of defined goals and exclusions.

…which requires constant monitoring and updating as technology and system access evolve.

…which requires a multi-step approach that uses process, policy, hardware, and software to protect your vital company data.

The Big Picture: Security Layers in an OpenEdge Application

TDE is part of OpenEdge's layered security strategy



PROGRESS

Who Needs Encryption?

Retail • Financial Services • Healthcare • Travel • Government • Online • Cloud • CRM

PIPEDA	Personal Information Protection and Electronic Documents Act
PCI DSS	Payment Card Industry Data Security Standard
HIPAA	Health Insurance Portability & Accountability Act
SOX	Sarbanes-Oxley Act
EU-DPD	European Union Data Protection Directive

Regulation forces compliance across markets.

Who Has the Pain?

Target Buyer	Operational Pain
CIO, VP or Technology Director	Legal and financial risks associated with lack of regulation compliance
Business Application Owner or Product Owner	Inability to comply with changing regulations
	Increasing need to protect and secure data for payment and information transactions
	Not being current with business requirements for processing credit card payments
IT Manager, Program Manager	High costs associated with the maintenance and management of older business applications
	Balancing system performance and encryption strength
	High cost of hardware redundancy

Data Security Options

- Use ABL Encryption Functions
- Encrypt Data Using O/S or SAN File Encryption
- 3rd-Party Encrypted SAN
- RDBMS Encryption

Option 1: Progress OpenEdge ABL Encryption Functions

- Operates at the field level
  - Requires significant rewrite and ongoing maintenance of existing code
- Does not encrypt the database
  - Poor performance- data not indexed, no range searches
  - Limited effectiveness from a security perspective - programmers put in position of “security risk”- mistakes, oversights, dishonesty can happen
  - May not pass auditor review
- Customer has to manage the encryption keys manually
- SQL reports do not decrypt values

Option 2: Encrypt Data Using O/S or SAN File System

- Performance is an issue – heavier overhead than DB encryption
  - Microsoft says file encryption is too slow for DB
- Security administrators must manually track the encryption keys for anything archived
- Security administrators cannot prevent the writing out of clear-text data
  - The DB and some OS utilities can write to other file systems that may not be encrypted

Option 3: 3rd-Party Encrypted SAN

- Same issues as file system: security of the data outside the secured environment not guaranteed
  - Backups, dumps, journal files, etc.
  - Anything “unencrypted in memory” can be written

Option 4: Database Encryption At Rest

- Database At-Rest (storage area level) Encryption
  - Data secure on-disk, backup, and dump
  - Data is unencrypted In-Memory = (up to) normal speed
- Separate but Secure Key Store and Key Management
- Policies control use of utilities
- Industry standard encryption routines supported
  - AES, DES, triple DES, etc.
- No application or code changes required!
- This is the solution chosen by most database vendors

This is the TDE option!

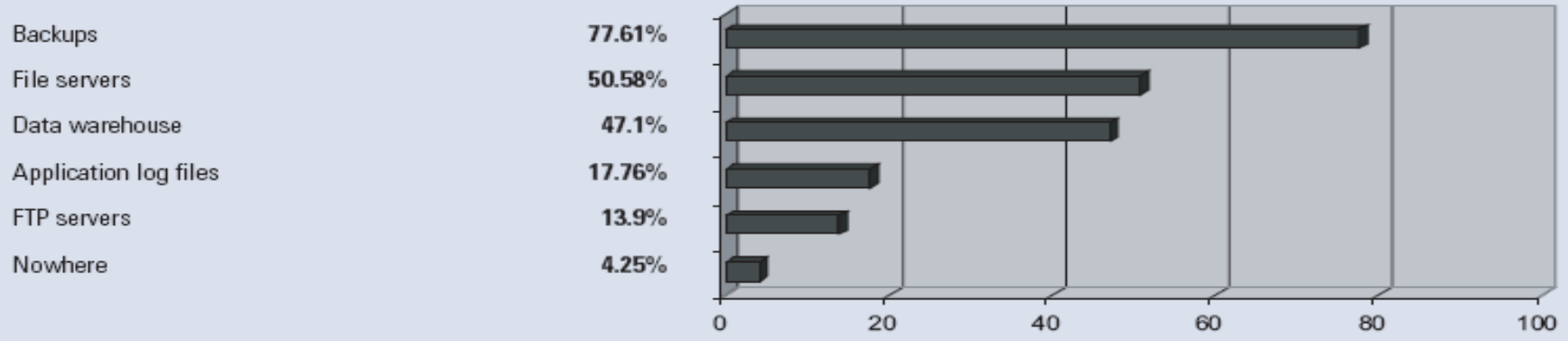
"Industry expectations are 'encryption at rest' because the major database vendors have proven this approach is performant, and less hassle than encrypting file systems."

—Carl G. Olofson,
IDC Analyst for Databases

Encryption – Industry View: Management

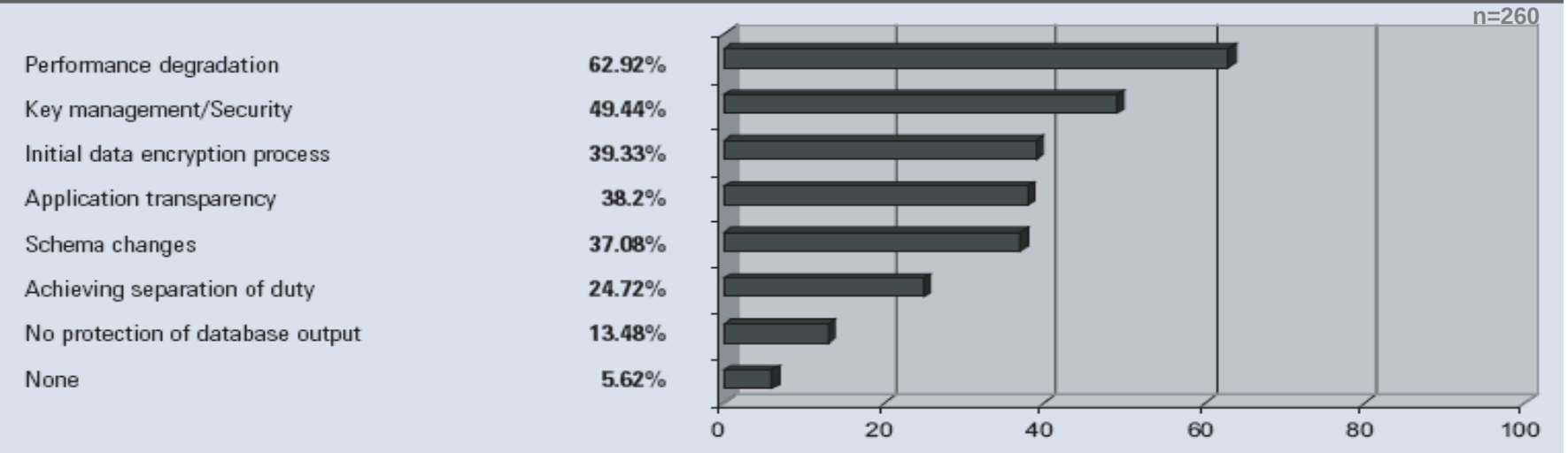
In addition to the database, where is sensitive information stored?

n=260



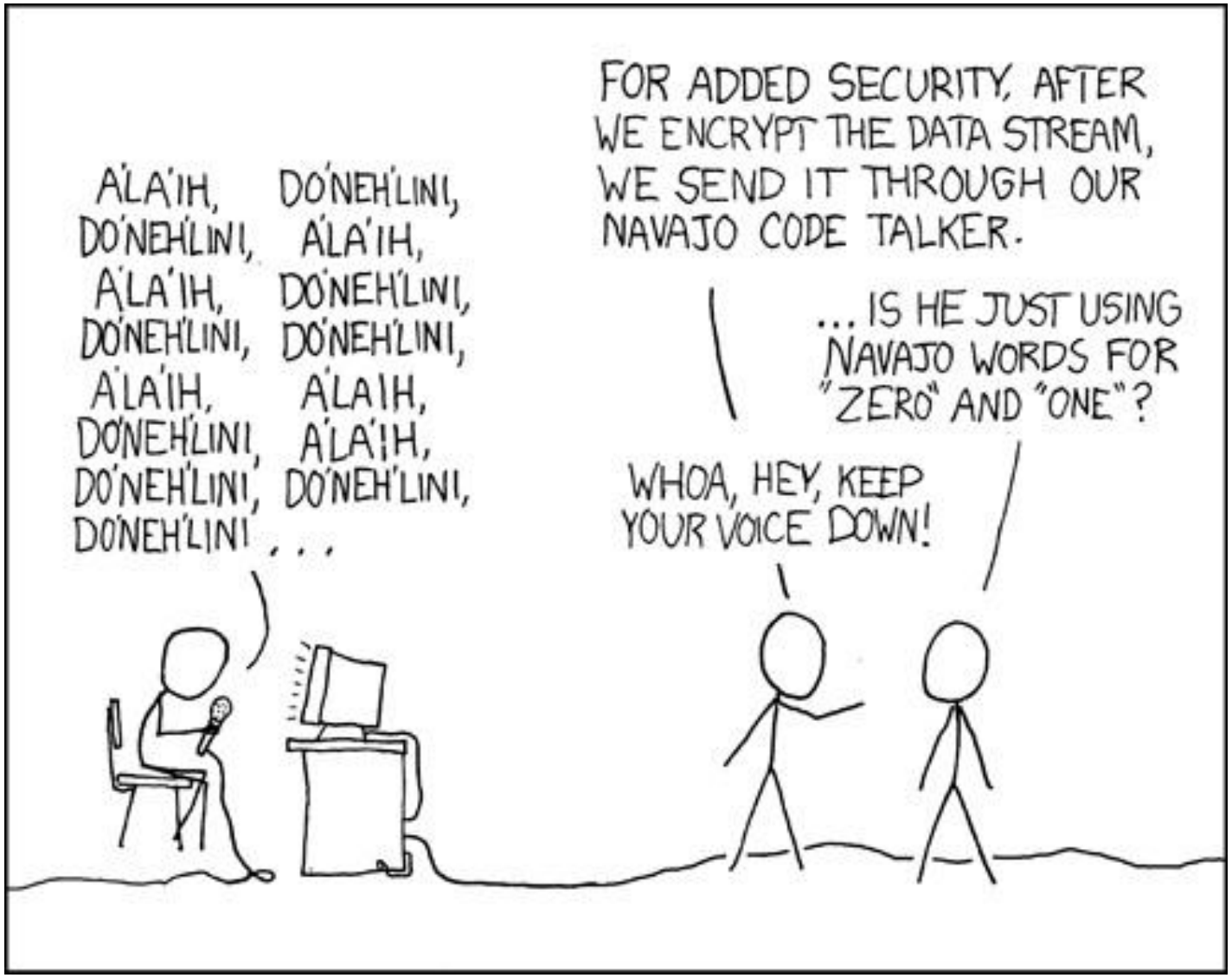
Biggest Challenges Associated with Database Encryption

n=260



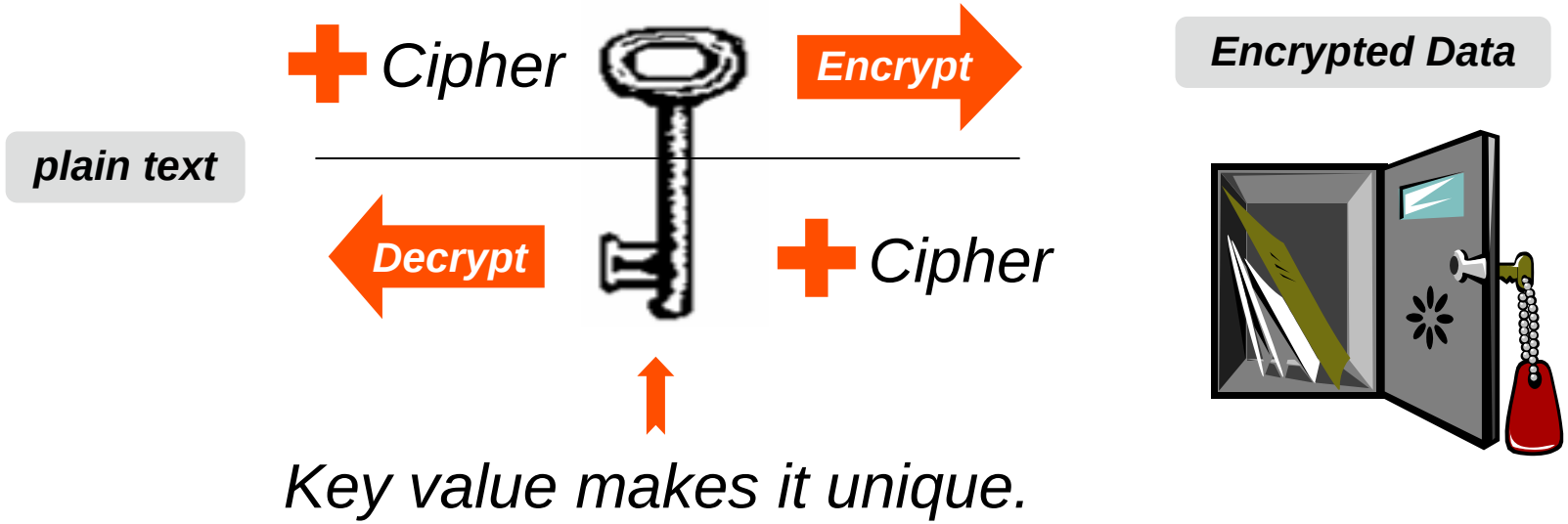
PROGRESS

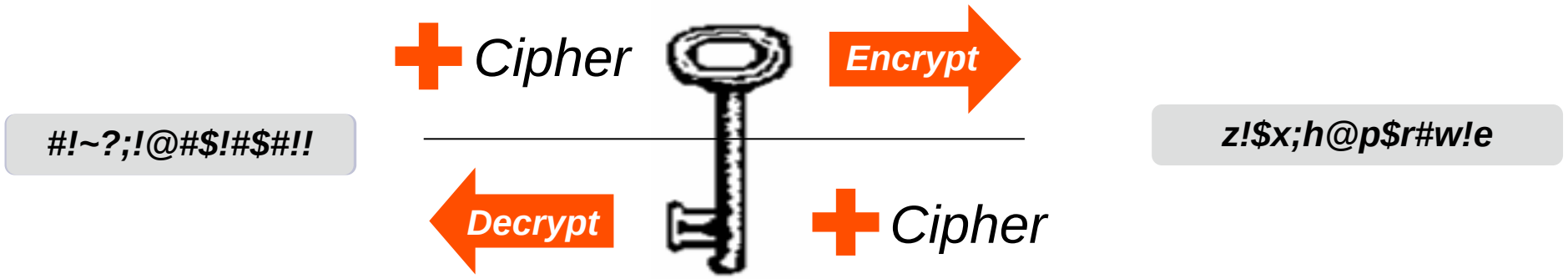
How Does Encryption Work?



Source: http://xkcd.com/257/

How Does Encryption Work?





Progress OpenEdge Transparent Data Encryption (TDE)

Transparent

- Encryption / Decryption is transparent to the application
- No need to move data or change code
- Full index query support

Data

- Provides data privacy while data is at rest
- Flexible: Selective encryption of Objects
- Storage engine encrypts database blocks on disk (access neutral)

Encryption

- Secure: Uses industry-standard encryption algorithms
- Provides secure encryption “Key Store”
- Limits access to physical data

How does Progress OpenEdge TDE Work?

Only 3 things you need to know to understand TDE

1. Products
2. Key Store
3. Encryption Policy

Thing 1: Products

Using TDE requires two installed products:

1. **OpenEdge Transparent Data Encryption**
   - First available in 10.2B
2. **OpenEdge Enterprise RDBMS**
   - NOT Workgroup!

Thing 2: The Key Store

The Most Critical Piece Of TDE

Stores the Database Master Key (DMK)

- Makes encrypted data unique

Unique per database

- File named: dbname.ks

Securing the DMK in the key store

- Stored separately from db
- Not part of database backup → Why not?
- Protected by passphrase based authentication

Data Object Encryption Keys

- Unique Keys for each db Object
  - If cracked, intruder only has access to that Object

Thing 3: Encryption Policies

Describes What And How To Encrypt

Policy Contents

- Object to encrypt
  - Table, Index, LOB (Type II)
  - Storage Area (Type I)
  - AI and BI recovery
- Cipher – algorithm & key size

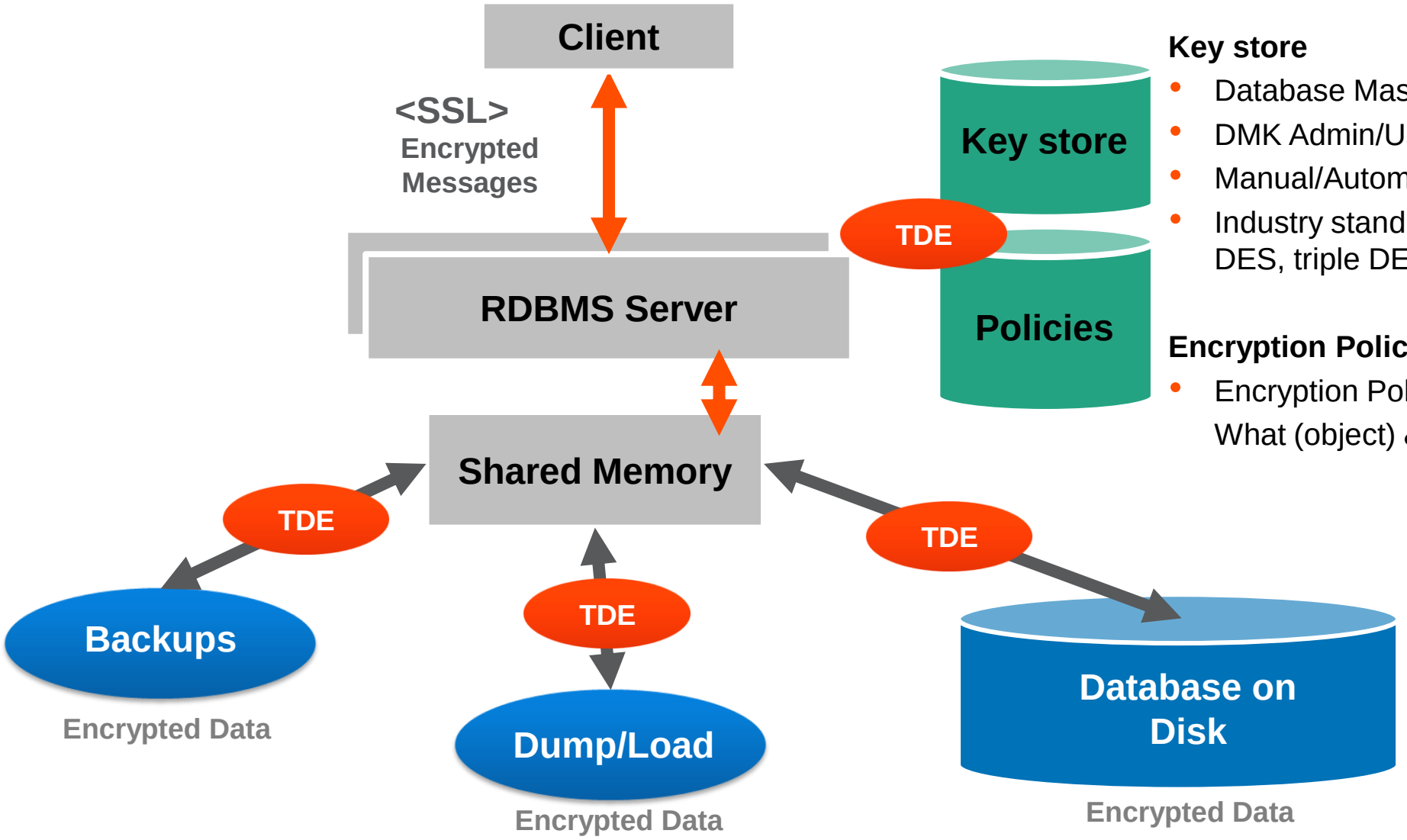
Secure (Key store administrator & DB administrator)

- Stored in “Encryption Policy Area”
- User prevented from direct record access

Policy Maintenance

- Add, remove, alter (cipher, key) online
- Epolicy tool, OpenEdge SQL, Data Admin tool

OpenEdge TDE Diagram



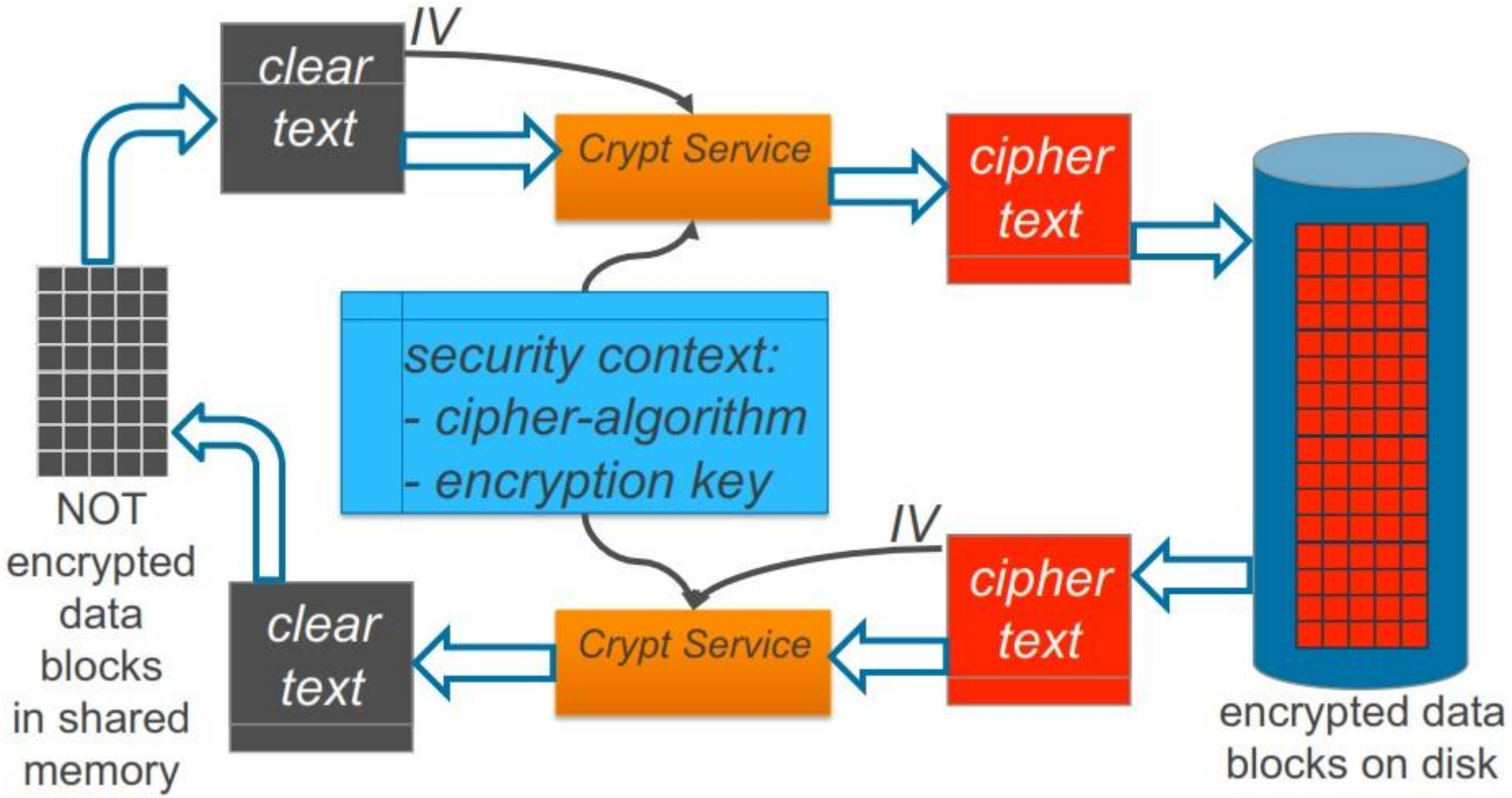
Key store

- Database Master Key (DMK)
- DMK Admin/User Passphrase
- Manual/Automatic Authentication
- Industry standard cipher algorithms AES, DES, triple DES, etc.

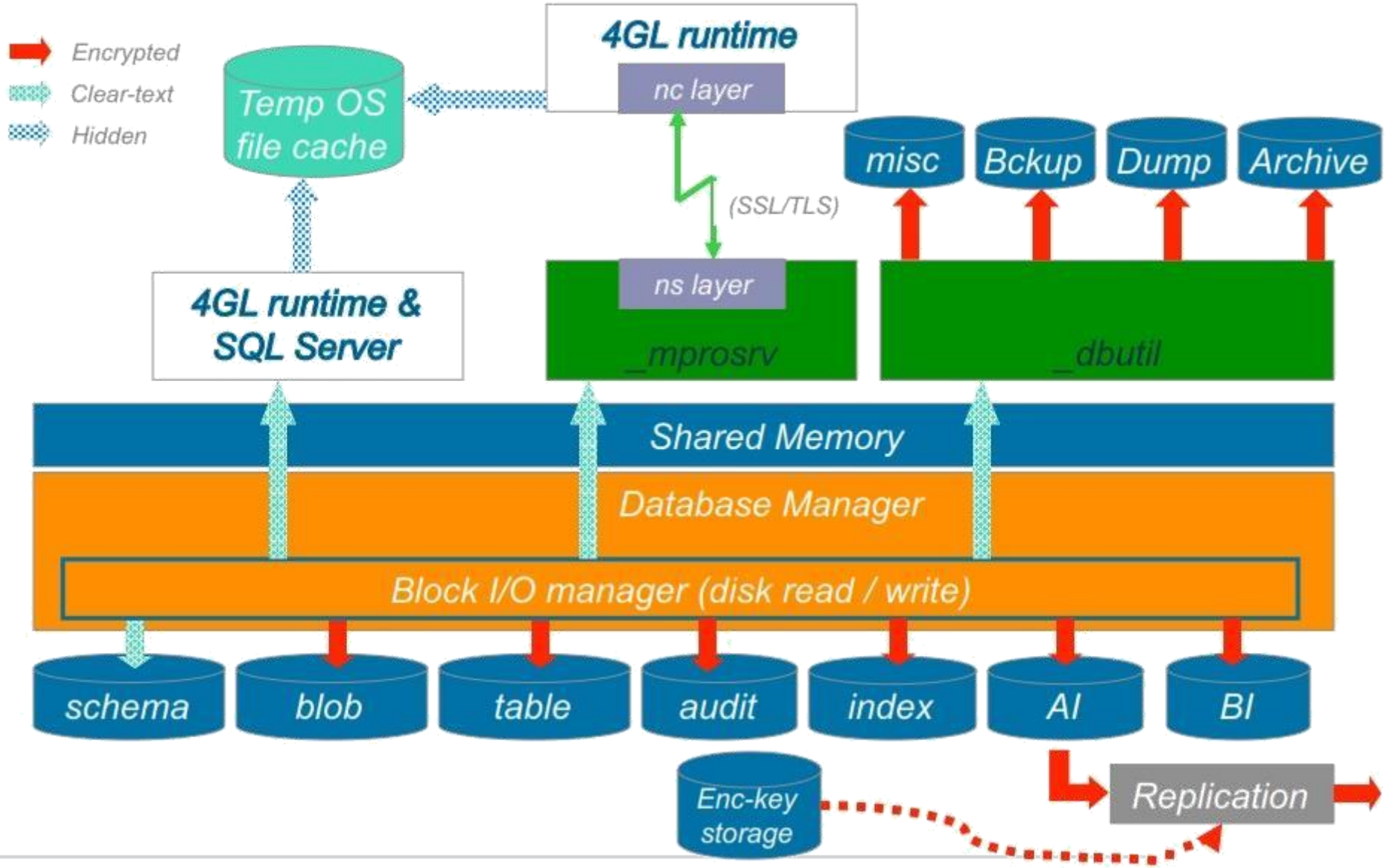
Encryption Policy Area

- Encryption Policies
  What (object) & how (cipher)

OpenEdge TDE – Block-Level Encryption



OpenEdge TDE – Encrypted Data Paths



Setting up Progress OpenEdge TDE

PROGRESS

Setting Up TDE – It's as easy as 1, 2, 3 (4)

1	*Add "Encryption Policy" Storage Area to the database*
2	*Enable the database for encryption*
3	*Configure encryption policies*
4	*Encrypt existing unencrypted data (optional)*

TDE Setup: Cheat Sheet

1 **Add the encryption policy area**

```
PROSTRCT ADD myDB encrypt_policy_area.st
```

2 **Enable TDE on the database**

```
proutil myDB -C enableencryption
```

3 **Add encryption policies for database objects**

```
proenv> proutil myDB
        -C epolicy manage area encrypt "DataArea100"
        -Passphrase
```

4 **Encrypt existing data (optional)**

Setting Up TDE – Deep Dive

1	Add “Encryption Policy” Storage Area to the database
2	Enable the database for encryption
3	Configure encryption policies
4	Encrypt existing unencrypted data (optional)

Setting Up TDE

1	**Add “Encryption Policy” Storage Area to the database**
2	*Enable the database for encryption*
3	*Configure encryption policies*
4	*Encrypt existing unencrypted data (optional)*

Encryption Policy Storage Area

Create a data area for encryption policies

- Type II area added to the database
- Name is "Encryption Policy Area"

Create structure definition file with policy area

```
e "Encryption Policy Area":12,64;8 .
```

Add the encryption policy area using PROSTRCT Add

```
PROSTRCT ADD mydb encrypt_policy_area.st
```

Policy area will normally not have much data in it.
One or two records per encrypted object

Setting Up TDE

1	*Add “Encryption Policy” Storage Area to the database*
2	**Enable the database for encryption**
3	*Configure encryption policies*
4	*Encrypt existing unencrypted data (optional)*

Enabling TDE

```
proutil db-name -C enableencryption
[-Cipher cipher-number] [-Autostart {user | admin}]
[-biencryption enable|disable]
[-aiencryption enable|disable]
[-Passphrase]
[[-userid userid][-password password]]
```

- Enables the database for TDE
  - Must be run on a command line
- Does not encrypt any data
- Creates the key store file

```
proutil tdeSport -C enableencryption
```

Database Key Store Built-in Accounts

Passphrase: A sequence of text, may include whitespace and punctuation, used to control access to a program or data such as an encryption key

Admin Account

- Must be used to change any key store value
- Used to administer off-line
  - Encryption configuration
  - Key store access
  - Manual/autostart mode

DBA Account

- Use for daily (non-encryption) admin tasks
- For example, use to start database servers and to access data

Recommendation: Use the admin account exclusively for administration

There are NO tools available from Progress to allow a key store file to be opened if the key store admin account passphrase is lost

Key Store Service Passphrase Delivery

Manual start mode

- Default mode
- More secure
- Requires a key store user passphrase every time the database is opened
- Can impact automated database tools
- Options:
  - Type in passphrase
  - Write ‘secure’ scripts to automate delivery of passphrase (very hard to do)

Autostart mode

- Less secure
- Automatically delivers account passphrase to open the key store
- Gives access to key store and data automatically
- Can be set to either key store account
- Account becomes default account for all users

Recommendation: Never turn on Autostart for a TDE database that may have a copy outside of the development lab

Setting Up TDE

1	*Add “Encryption Policy” Storage Area to the database*
2	*Enable the database for encryption*
3	**Configure encryption policies**
4	*Encrypt existing unencrypted data (optional)*

Encryption Policies

Encryption attributes of database objects are managed through encryption policies

Policies are stored in the Encryption Policy Area

To administer policies you must be a ***DBA*** and have access to the ***key store admin account***

Built-in to TDE security protects policy records

Access requires command be run locally

Encryptable Database Objects

OpenEdge Database

Type I data area

Entire area encrypted

Tables

Indexes

LOBs

Cannot be encrypted

- *Schema Area*

Type II data area

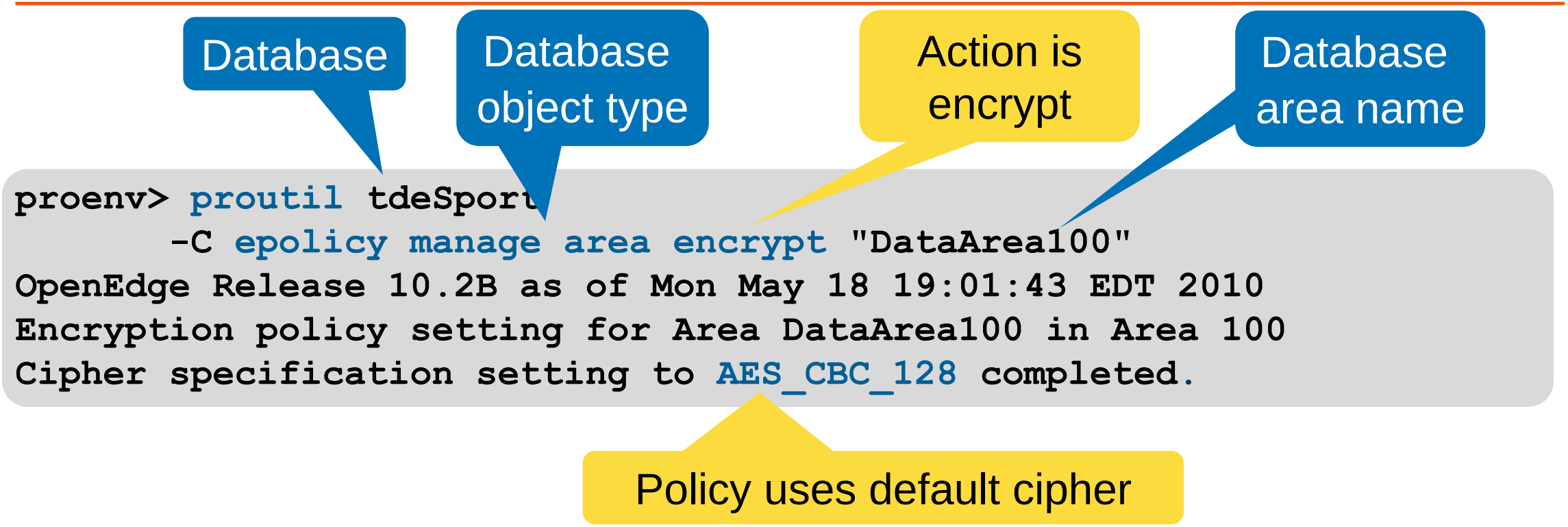
Selected objects encrypted

Table	Index	LOB
Index	LOB	Table
Index	Table	LOB
Index	LOB	Table

Cannot be encrypted

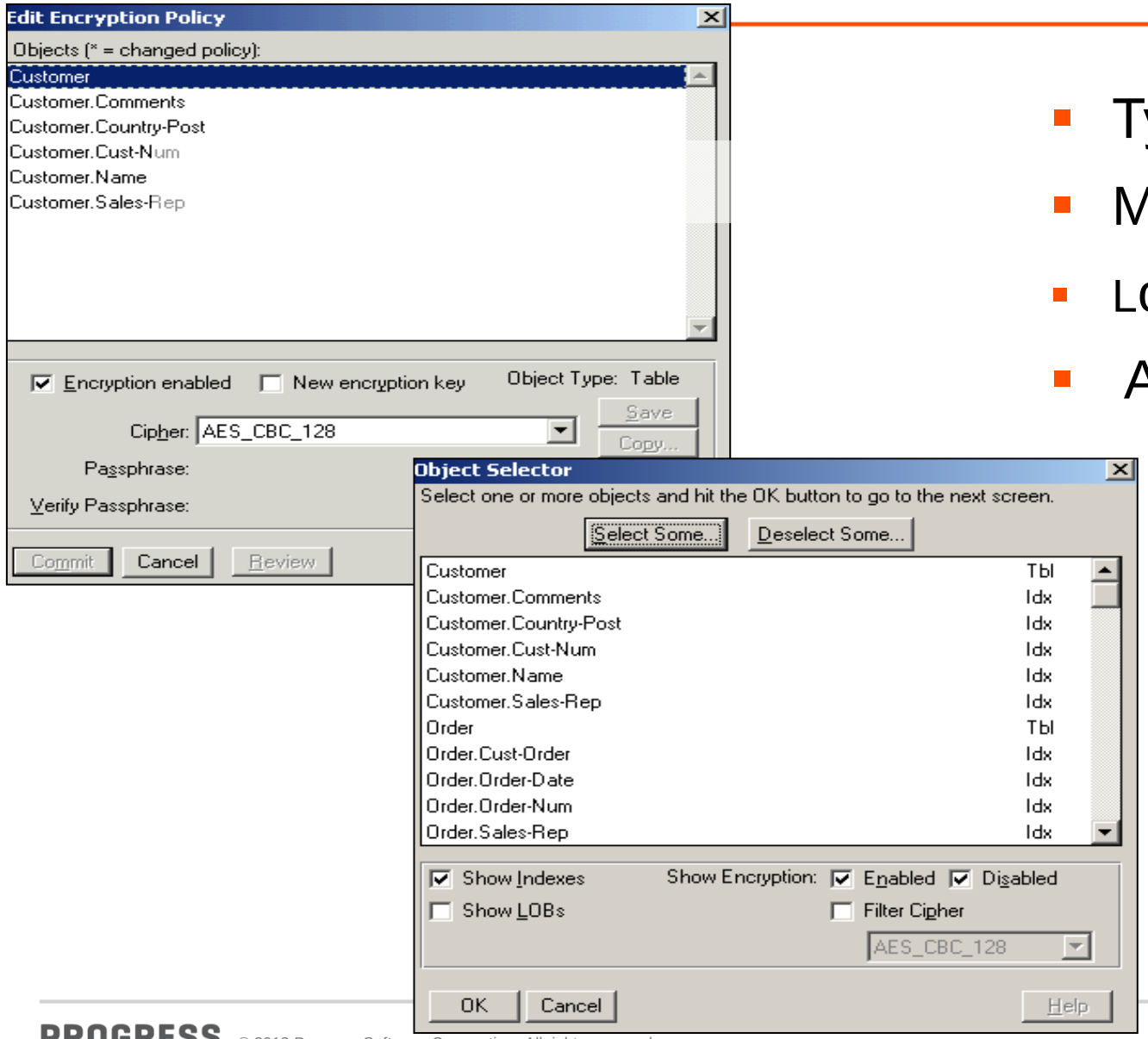
- *Encryption Policy Area*

Creating an Encryption Policy



Putting the pieces together:
The policy for this data base object is created and placed in the encryption policy storage area of the database

Setting Policy with Data Admin Tool



- Type II "PUB" schema only
- Multi select UI
- Local access only
- Admin
  - Security
    - Encryption Policies
      - Edit Encryption Policies . . .

Select the Right Cipher
Based on the Value of the Data

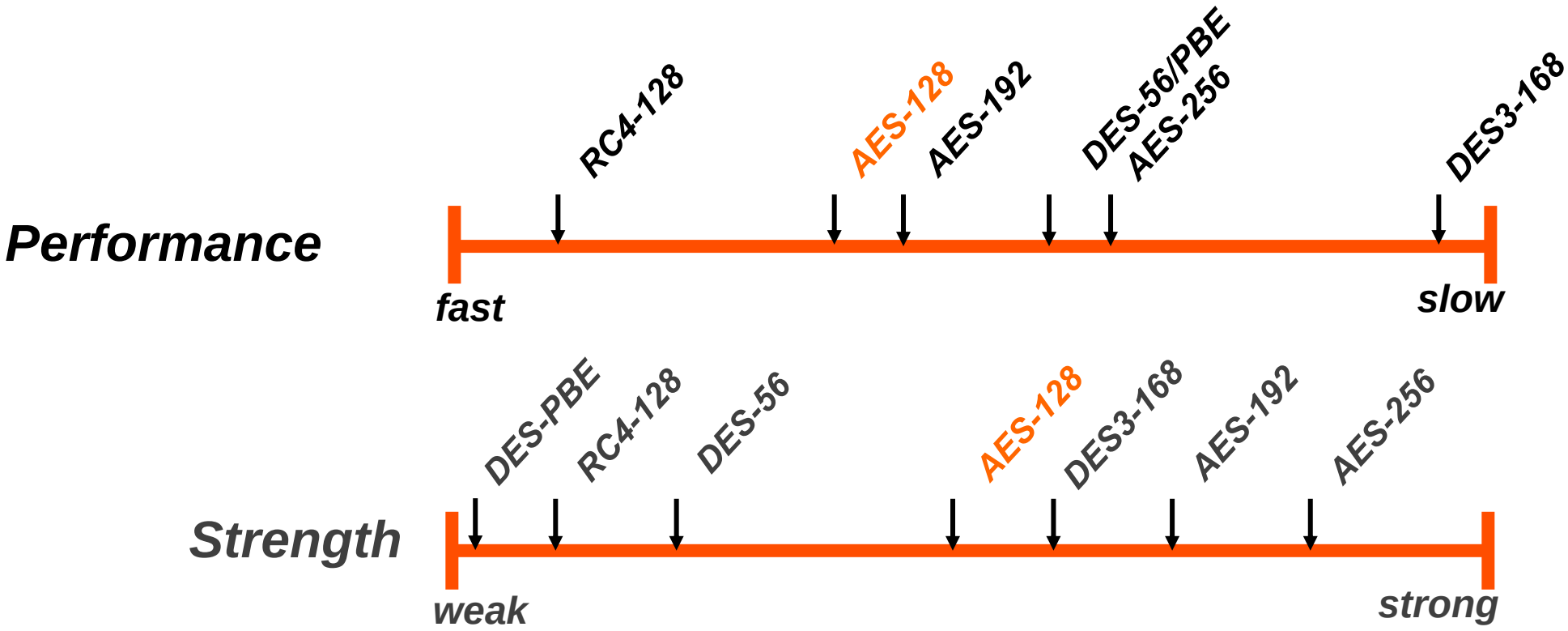
Considerations when selecting a cipher:

- Is it strong enough to provide desired security?
- Is it fast enough for the applications requirements?

The strength is based on three factors

- *Algorithm type – mathematical formula*
- *Mode - used to manipulate the key data*
- *Key size – In bits*

Balance strength against performance

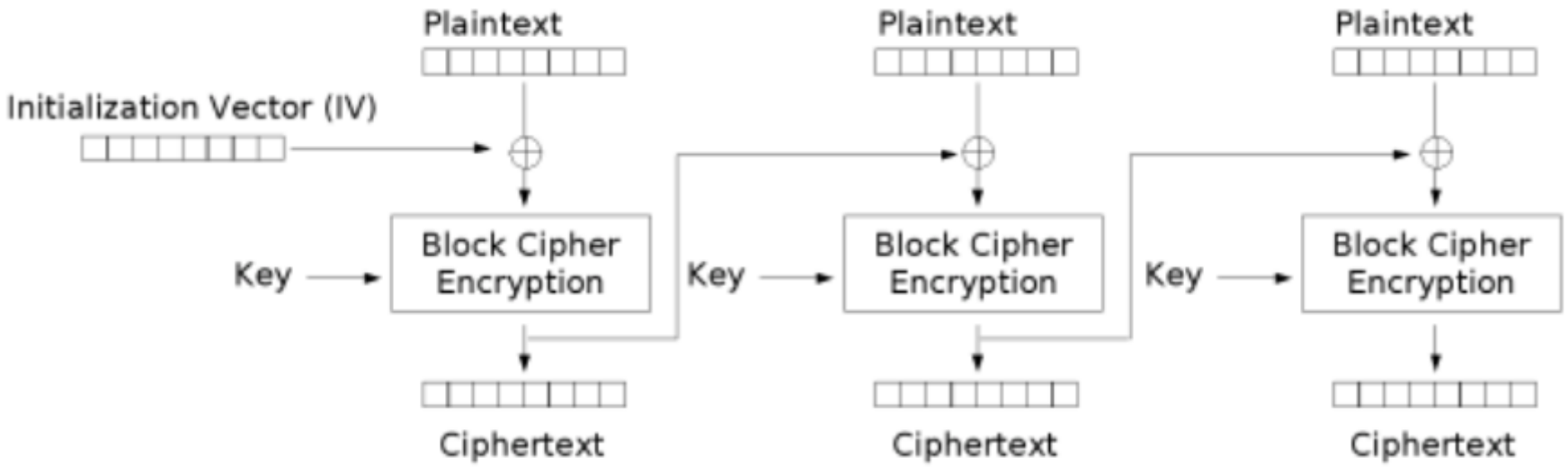


What Does “AES_CBC_128” mean?

AES = the "Advanced Encryption Standard" encryption algorithm

CBC = Cipher Block Chaining encryption mode

128 = length of encryption block and key in bits (16 bytes)



Cipher Block Chaining (CBC) mode encryption

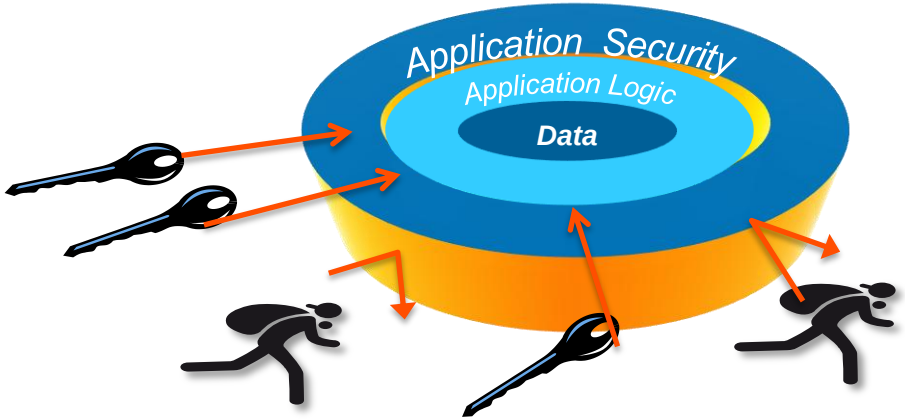
Setting Up TDE

1	*Add “Encryption Policy” Storage Area to the database*
2	*Enable the database for encryption*
3	*Configure encryption policies*
4	**Encrypt existing unencrypted data (optional)**

Options for Encrypting Existing Data

1	*Data are encrypted, when updated, by the* **normal course of database updates** *each time a block is written to the database*
2	**Dump and load data** *objects, encrypting data during the load operation*
3	*Run* **EPOLICY MANAGE UPDATE** *command to encrypt all data in a database object*

How do you know what data are encrypted, and what are not ????



Viewing Database Object Encryption Status

- Provides information on the encryption policy for the selected database object

```
proenv> proutil t1demo -C epolicy scan
                area "DataArea101"
OpenEdge Release 10.2B1P as of Thu Oct 29 …
AREA DataArea101 /
101 CURRENT AES_CBC_128 V:0 200 of 627 blocks encrypted
```

Number of blocks encrypted

Total number of blocks

Encrypting Data

Encrypts all blocks in the database object that are not already encrypted using the current policy

Action is update

```
proenv> proutil t1demo -C epolicy manage
                area update "DataArea101"
OpenEdge Release 10.2B1P as of Thu Oct 29 19:01:53 EDT 2010
AREA DataArea101 /
101 CURRENT AES_CBC_128 V:0 427 of 627 blocks encrypted
```

Number of blocks encrypted

Total number of blocks

Encryption Policy Reports

- Quick Encryption Policies report
  - Shows current cipher name and policy version
- Detailed Encryption Policies report (shown)
  - Information similar to Detailed Table report, but includes encryption information

```
Reporting only objects with encryption enabled at the object level
================================================================================
============================== Table: Customer ===============================
 Object Name : Customer
 Object Type : Table
 Storage Area: Customer/Order Area

      Policy Version Cipher Name          Policy State
      -------------- -------------------- --------------------
                   1 AES_CBC_128          Current
                   0 AES_CBC_256          Previous

 Object Name : Comments (Table: Customer)
 Object Type : Index
 Storage Area: Customer Index Area
        No policy information available for object.
```

Policies version

Current and Previous policies

Connecting to TDE Enabled Databases

- You can supply a passphrase using
  - -Passphrase for commands
  - -KeyStorePassPhrase on the ABL CONNECT statement
    - Can only be used on a local, single-user connection
- Use with manual mode or to override autostart mode

```
> proserve myDB 1234 -Passphrase
Please enter the Passphrase for database myDB
```

```
CONNECT myDB -1 -KeyStorePassPhrase VALUE(QUOTER(myVar))
```

Recommendation: Create a dialog box to prompt for the passphrase prior to CONNECT statement and do not echo the characters

Connecting to TDE Enabled Databases

- No passphrase is needed when connecting to a database server using a client-server or self-service client if the server is already started
  - Virtual encryption keys are securely pre-loaded and available to decrypt and encrypt data in the database

TDE Environment: Temporary Files

Both ABL and OpenEdge SQL clients create temporary storage files when accessing databases

-t startup parameter (save temp files)

- You cannot connect when an ABL client uses the -t parameter
- Using OpenEdge SQL client the -t startup parameter is ignored

In a TDE database temporary files:

- Are hidden and readable (not encrypted and may be read)
- Are forcibly removed when a 10.2B client process ends

Other Things That ARE Encrypted

Data automatically encrypted

- PROBKUP
- After image hot-standby databases
- OpenEdge Replication targets
  (some setup required)

Data optionally encrypted

- Binary dump and load
- Audit archive and load

Recommendation: Backup the database and the key store to different media

A Few Final Comments…

Maintaining TDE Enabled Databases

Modifying a virtual data encryption keys

```
PROUTIL dbname -C epolicy manage
        object-type rekey object-name
```

Changing the cipher of an encrypted database object

```
PROUTIL dbname -C epolicy manage
        object-type cipher object-name
        -Cipher cipher-num
```

TDE Environment

- The Alternate Buffer Pool
  - A second shared-memory resident buffer pool, just like the one you are already used to
  - Set size with –B2
  - Only objects you specify are cached there

- Policy cache buffer
  - -ecsize

Testimonial from Fiserv – a TDE user

Benefits

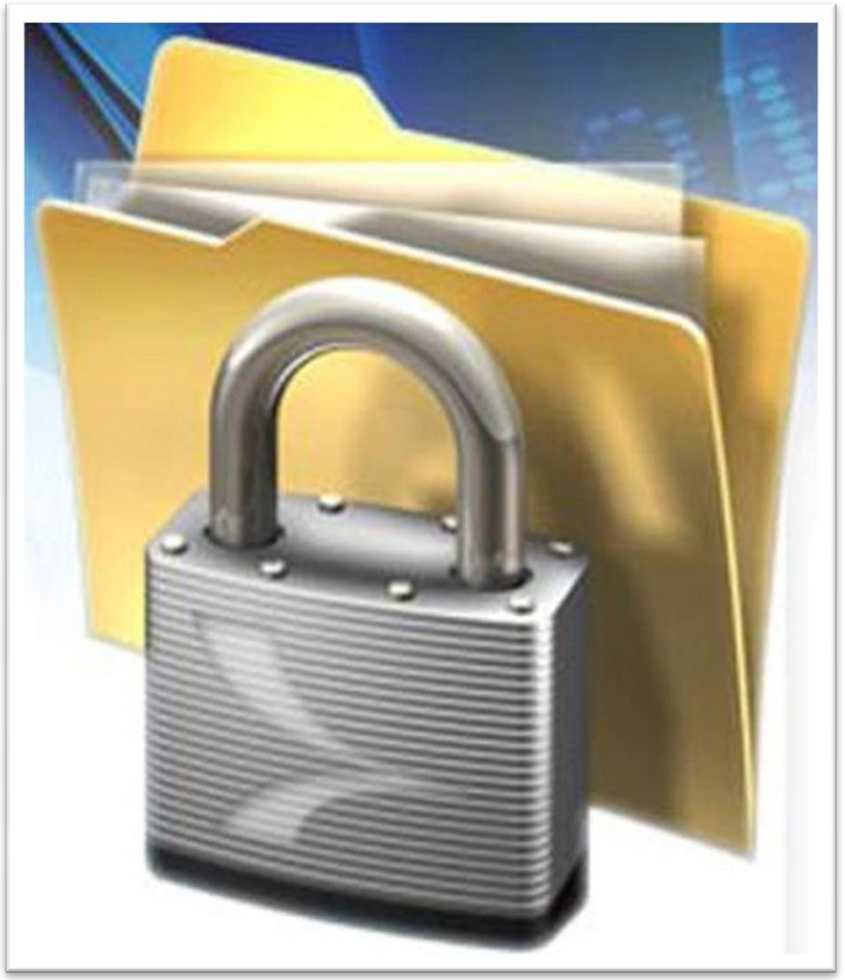
- TDE will ensure data privacy across the entire lifecycle
- Maintain competitive advantage and ability to interface with third parties by adhering to PCI DSS
- Increased IT performance will save time and reduce costs

"We always try to improve our performance and get things to run faster. We tested a fully encrypted database and there was only a 4% decrease in performance versus an unencrypted database. We tested that with alternative data pools, we actually gained back almost 2% of that initial performance degradation. We believe with additional fine tuning the performance will continue to improve."

Progress OpenEdge TDE – Top 10

Transparent Data Encryption

- Advanced encryption approach for database
- Protects data at the table/index level
- Requires no application changes
- All application features work as before
- Very low impact on performance (< 5%)
- Broad applicability to many use-cases
- One important aspect of an overall security strategy
- Available as of OpenEdge release 10.2b
- Best thing since sliced bread
  - To use TDE you need two OpenEdge products:
    - Enterprise OpenEdge Database
    - Transparent Data Encryption



Questions?

PROGRESS