Data Privacy and Software Development

Thirty years ago, data privacy meant making sure there was no unauthorized access to payroll records. Even an attempted or accidental unauthorized access was grounds for immediate dismissal. Today it means much more than the embarrassment of knowing the salary of a co-worker or even a top corporate executive. Information is a corporate asset as well as a liability. Possibilities for unscrupulous uses of information abound. Imagine what an unprincipled competitor could do with any one of the following:

- A list of your current and prospective customers with names, contacts, and phone numbers
- Your customer buying patterns - products, sales forecasts, geographic trends
- Your product plans and directions
- A list of your employees, their job titles, phone numbers and responsibilities

The liabilities are at least as severe. Imagine what a mischievous third party could do with any one of the following:

- Patient list, complete with diagnosis, prognosis, and proscribed treatments
- Delivery schedules, routes, and carriers for hazardous materials
- Payment dates, account numbers, and addresses of investors

Global Privacy Mandates

United States
- Health Insurance Portability and Accountability Act (HIPAA), 1996
- Gramm, Leach, Bliley Act, 1999

Australia
- Privacy Amendment Act, 2000

Canada
- Personal Information Protection and Electronic Documents Act

Hong Kong
- Principle 4 Security of Personal Data

Japan
- Privacy of Personal data

New Zealand
- Principle 5 Security of Personal Information

United Kingdom
- Data Protection Act, 1998

European Union
- European Union Council Personal Data Protection Directive

Most companies have already used security software to address many of these issues. Whether it is passwords protected with RACF, ACF2, Top-Secret or even Windows, most access to production systems is authorized. Employees have access to the data to complete their jobs and little more. Super-sensitive data in production is encrypted.

As more and more business processes are automated, the need for secure information has grown. Governments worldwide are getting involved.

Software Development Process

Through legislation and good business practices, desire for data privacy has expanded to include the software development process itself. Although software engineers are an honorable lot, they don't require access to this information – or do they? What about a simple production system failure? What about system test data that is currently a sampling of the data in production? How can software problems be resolved without the data that caused them to malfunction? How can systems be tested without representative test data?

Problem Resolution and Other Software Testing

The only reliable way to solve a software problem is to be able to repeat it. In an ideal world, we would be able to recreate the problem with the live system using some set of test cases we already had. If not, we could make some educated guesses as to the cause of the problem and attempt to recreate it through trial and error using our test cases and modifications to them. This, as it sounds, can be a time consuming and expensive process. To be expedient, we learned to get the data required for the malfunctioning process from the live system and put it in the

test environment for problem resolution. Using this production data, however, violates the tenets of both the legislation and good data privacy practices.
As it turns out, the specifics of who and what are far less important in problem resolution of this kind than simply the data structures and usually numeric values that caused the problems. It doesn't matter to the application or the software engineer whether it was the president of the company whose salary caused the payroll application to malfunction or whether it was the janitor's. It doesn't matter that invoice 27 died because of an order for nuclear bomb triggers or an order for widgets.

This is true for all kinds of software testing – unit testing, string testing, system testing, acceptance testing, volume and performance testing. Software and software testing doesn't care about who, what, when and where of the live production values, but rather the size of the data values and the combinations of events. Data stripped of its meaning to the real world can be just as useful in performing software tests and resolving problems.

Data Masking

Data that has been stripped of its real world meaning has many names. Some refer to it as Masked, Jumbled, Encrypted, Scrubbed, or even Sanitized. Many of these names come from the processes used to make the data void of its real world meaning, but still useful in software testing. This transformation process has several challenges:

- **Consistency** - making sure that the same data is transformed in a consistent way so its relationships are maintained. If you change a customer's name or an address in one place you want to be sure it is changed in the same manner in all places that refer to that customer. If you change the employee number in one place, you want to make sure it is changed to the same value for the same employee in other places.
- **Uniqueness Constraints** - masked data must support uniqueness constraints. If you change a social security number used as a unique key, not only does it have to be changed to the same value everywhere it is referenced, but it also should maintain the uniqueness property with respect to other social security numbers in the system.
- **Referential Integrity** - even well-known system assigned values may have to be changed but in a consistent way. Product numbers, Customer numbers, account numbers, are examples. Any data value used as a primary key or a foreign key must be changed everywhere it is used in a consistent way to maintain the referential integrity of the data.
- **Authorization** - only authorized personnel should be allowed to define the data transformations and all data extracted from live systems must be forced to go thru the appropriate transformations unseen in its native form. Many companies have written entire applications to perform these functions against data in their live systems as a matter of expediency. This was a major undertaking and still required authorized personnel.

SoftBase's TestBase

TestBase is a comprehensive test data management tool. One of its features, Data Population, addresses the issues of data privacy head on by using the following techniques:

- Extract referentially intact sets of data from live production environments
- Mask data as it is extracted using 7 techniques
  - Select data values from any VSAM or QSAM file
  - Select data values from any DB2 table or view
  - Select data values from values supplied with the product for names, addresses, phone numbers, etc. in sequential or random fashion

- Select data values described by an SQL select statement (TestBase View)
- Select random values
- Jumble numeric or character positions
- Transformation tables to lookup real values and supply result values

➢ Generate test data from scratch for a single table or dataset or a family of referentially related ones
➢ Guarantee only authorized personnel can define the data masking rules
➢ Guarantee all extracts of live information have the rules applied

Summary

Whether your organization falls under a regulatory directive, is seeking to reduce potential liabilities, or simply wants to protect its competitive advantage by keeping a tight lid on operational data, the SoftBase Db2 tools can support you through the planning, data manipulation, testing, and deployment phases. In addition, the proven capabilities of TestBase can simplify each phase of the implementation, reducing the time required and improving the integrity and reliability of the new system once it is deployed.