

MỤC LỤC

MỤC LỤC	i
DANH MỤC TỪ VIẾT TẮT	vi
DANH MỤC BẢNG	vii
DANH MỤC HÌNH VẼ	viii
MỞ ĐẦU	1
CHƯƠNG 1. TỔNG QUAN VỀ AN TOÀN	2
CƠ SỞ DỮ LIỆU.....	2
1.1 GIỚI THIỆU.....	2
1.2 MỘT SỐ KHÁI NIỆM CƠ BẢN.....	5
1.2.1 Hệ cơ sở dữ liệu.....	5
1.2.1.1 Dữ liệu	5
1.2.1.2. Phần cứng	6
1.2.1.3 Phần mềm	6
1.2.1.4 Người dùng	7
1.2.1.5 Thẻ hiện	8
1.2.1.6 Lược đồ.....	8
1.2.2 Thiết kế cơ sở dữ liệu	8
1.2.3 Các mức mô tả dữ liệu.....	11
1.2.4 Ngôn ngữ SQL	12
1.3 CÁC HIỂM HỌA VÀ TẤN CÔNG ĐỐI VỚI CƠ SỞ DỮ LIỆU	12
1.3.1 Hiểm họa ngẫu nhiên và có chủ ý	13
1.3.2 Tấn công bên trong và bên ngoài	13
1.3.3 Mười hiểm họa hàng đầu tới an toàn cơ sở dữ liệu	15
1.4 CÁC YÊU CẦU BẢO VỆ CƠ SỞ DỮ LIỆU	26
1.4.1 Bảo vệ chống truy nhập trái phép	26
1.4.2 Bảo vệ chống suy diễn	27
1.4.3 Bảo vệ toàn vẹn cơ sở dữ liệu	27
1.4.4 Khả năng lưu vết và kiểm tra	28

1.4.5 Xác thực người dùng.....	28
1.4.6 Quản lý và bảo vệ dữ liệu nhạy cảm.....	28
1.4.7 Bảo vệ nhiều mức	29
1.4.8 Sự hạn chế.....	29
1.5 CÂU HỎI ÔN TẬP	30
CHƯƠNG 2. CÁC MÔ HÌNH VÀ CHÍNH SÁCH AN TOÀN	31
2.1 MỘT SỐ KHÁI NIỆM CƠ BẢN.....	31
2.1.1 Mô hình an toàn	31
2.1.2 Chính sách an toàn	31
2.1.3 Quy tắc trao quyền	32
2.2 CÁC MÔ HÌNH AN TOÀN CƠ SỞ DỮ LIỆU	34
2.2.1 Kiểm soát truy nhập trong các hệ thống hiện tại	34
2.2.2 Các mô hình an toàn tùy ý	35
2.2.2.1 Kiểm soát truy nhập tùy ý	35
2.2.2.2 Hạn chế của kiểm soát truy nhập tùy ý	53
2.2.3 Các mô hình an toàn bắt buộc.....	55
2.2.3.1 Kiểm soát truy nhập bắt buộc	57
2.2.3.2 Mô hình an toàn dữ liệu quan hệ đa mức	62
2.2.3.3 Hạn chế của kiểm soát kiểu bắt buộc	66
2.2.4 Các mô hình an toàn khác	68
2.3. CÂU HỎI ÔN TẬP	68
CHƯƠNG 3. AN TOÀN TRONG DBMS	70
3.1 THIẾT KẾ DBMS AN TOÀN.....	70
3.1.1. Các cơ chế an toàn trong các DBMS	72
3.1.2 Các kiến trúc của DBMS an toàn.....	77
3.1.2.1 Kiến trúc chủ thể tin cậy.....	79
3.1.2.2. Các kiến trúc Woods Hole.....	81
3.2 GIỚI THIỆU MỘT SỐ HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU	96
3.2.1 Sơ lược kiến trúc một số hệ quản trị	96

3.2.1.1 Kiến trúc của Oracle	98
3.2.1.2 Kiến trúc của MySQL	105
3.2.1.3 Kiến trúc của SQL Server.....	111
3.2.2 Một số lỗ hổng trong các hệ quản trị	116
3.2.2.1 Một số lỗ hổng trong Oracle.....	117
3.2.2.2 Một số lỗ hổng trong MySQL	120
3.2.2.3 Một số lỗ hổng trong SQL Server	122
3.3 CÁC VẤN ĐỀ AN TOÀN CHUNG TRONG DBMS	123
3.3.1 Xác thực (Authentication).....	124
3.3.1.1 Xác thực mức hệ điều hành	125
3.3.1.2 Xác thực mức cơ sở dữ liệu.....	125
3.3.1.3 Xác thực mạng hoặc bên thứ ba	126
3.3.1.4 Các thành phần xác thực của nhà cung cấp cơ sở dữ liệu	127
3.3.1.5 Chính sách mật khẩu	129
3.3.2 Cấp quyền (Authorization)	130
3.3.3 Kiểm toán (Auditing).....	132
3.4 CÁC CƠ CHẾ AN TOÀN TRONG HỆ QUẢN TRỊ ORACLE	136
3.4.1 Cơ sở dữ liệu riêng ảo (VPD)	137
3.4.1.1 Tổng quan về VPD	137
3.4.1.2 Các thành phần của chính sách VPD.....	138
3.4.1.3 Cấu hình một chính sách VPD	139
3.4.2 An toàn dựa vào nhãn trong Oracle	141
3.4.2.1 Tổng quan về OLS.....	141
3.4.2.2 Nhãn dữ liệu và nhãn người dùng	142
3.4.2.3 Cấu hình một chính sách OLS	144
3.4.2.4 Sử dụng VPD để thi hành chính sách OLS	145
3.4.3 Cơ chế Kiểm toán mịn (Fine-grained auditing).....	146
3.4.4 Các cơ chế an toàn khác.....	148
3.4.4.1 Oracle Advanced Security	149

3.4.4.2 Oracle Secure Backup (OSB)	151
3.5 CÂU HỎI ÔN TẬP	151
CHƯƠNG 4. AN TOÀN ỦNG DỤNG.....	152
4.1 GIỚI THIỆU.....	152
4.2 VÂN ĐỀ QUẢN LÝ TÀI KHOẢN VÀ MẬT KHẨU NGƯỜI DÙNG	153
4.2.1 Lỗ hổng trong quản lý mật khẩu cơ sở dữ liệu	154
4.2.2. Kiểm soát việc truy nhập vào CSDL	159
4.3 XÁO TRỘN MÃ ỦNG DỤNG	165
4.3.1 Phân tích điểm yếu mã nguồn và giả mã	166
4.3.2 Tiền biên dịch và mã hóa mã ứng dụng	177
4.4 BẢO VỆ CƠ SỞ DỮ LIỆU KHỎI CÁC TẤN CÔNG SQL INJECTION ..	179
4.4.1 Hiểu biết về SQL injection	180
4.4.2 Quản lý, giám sát, cảnh báo và ngăn chặn.....	189
4.5 CẢNH GIÁC TRƯỚC SỰ KẾT HỢP GIỮA LỖ HỒNG SQL INJECTION VÀ TRÀN BỘ ĐÊM.....	199
4.6 LỐP ỦNG DỤNG TRÊN MÁY CHỦ.....	201
4.7 BỘ ĐÓNG GÓI ỦNG DỤNG	202
4.8 HƯỚNG TÓI LIÊN KẾT GIỮA MÔ HÌNH NGƯỜI SỬ DỤNG VÀ MÔ HÌNH CƠ SỞ DỮ LIỆU NGƯỜI DÙNG	204
4.9 TÓM TẮT	205
4.10 CÂU HỎI ÔN TẬP	205
CHƯƠNG 5. AN TOÀN TRONG CƠ SỞ DỮ LIỆU THỐNG KÊ	206
5.1 GIỚI THIỆU.....	206
5.1.1 Dạng biểu diễn của cơ sở dữ liệu thống kê.....	207
5.1.2 Các cơ sở dữ liệu thống kê trong thực tế	209
5.1.3 Vấn đề bảo vệ cơ sở dữ liệu thống kê	211
5.2 CÁC ĐẶC TRƯNG CỦA CƠ SỞ DỮ LIỆU THỐNG KÊ.....	213
5.3 CÁC KHÁI NIỆM CƠ BẢN	214
5.3.1. Lộ chính xác và lộ từng phần.....	214
5.3.2 Kiến thức làm việc và kiến thức bổ sung.....	215

5.3.3 Công thức đặc trưng.....	216
5.3.4 Tập truy vấn	217
5.3.4 Các truy vấn thống kê	218
5.3.5 Khái niệm bậc	219
5.3.6 Thống kê nhạy cảm	219
5.4 CÁC TẦN CÔNG VÀO CƠ SỞ DỮ LIỆU THỐNG KÊ	219
5.5 CÁC KỸ THUẬT CHỐNG SUY DIỄN	220
5.5.1 Các kỹ thuật khái niệm	222
5.5.1.1 Mô hình lưới	222
5.5.1.2 Phân hoạch khái niệm.....	229
5.5.2 Các kỹ thuật hạn chế	231
5.5.2.1 Kiểm soát kích cỡ tập truy vấn.....	232
5.5.2.2 Kiểm soát kích cỡ tập truy vấn mở rộng	241
5.5.2.3 Kiểm soát chồng lấp tập truy vấn (query-set overlap control)	244
5.5.2.4 Kiểm toán	245
5.5.2.5 Gộp (Microaggregation)	246
5.5.2.6 Giấu ô	248
5.5.2.7 Kỹ thuật kết hợp	251
5.5.3 Các kỹ thuật gây nhiễu.....	255
5.5.3.1 Kỹ thuật gây nhiễu dữ liệu	255
5.5.3.2 Kỹ thuật gây nhiễu đầu ra.....	271
5.6 SO SÁNH CÁC KỸ THUẬT CHỐNG SUY DIỄN	274
5.7 CÂU HỎI ÔN TẬP	281
TÀI LIỆU THAM KHẢO	281
PHỤ LỤC: Tìm hiểu cơ sở dữ liệu thống kê của Tổ chức Thương mại thế giới (WTO)	283

DANH MỤC TỪ VIẾT TẮT

CSDL	Cơ sở dữ liệu
DBMS	Hệ quản trị cơ sở dữ liệu
OS	Hệ điều hành
System-R	Hệ thống R
SDB	Cơ sở dữ liệu thống kê
Tracker	Tần công trình theo dõi
SQL	Ngôn ngữ truy vấn có cấu trúc
MAC	Chính sách kiểm soát truy nhập bắt buộc
DAC	Chính sách kiểm soát truy nhập tùy ý
RBAC	Chính sách kiểm soát truy nhập dựa vào vai trò

DANH MỤC BẢNG

Bảng 2.1 Ma trận quyền.....	34
Bảng 2.2: Cơ sở dữ liệu quan hệ EMPLOYEE	48
Bảng 2.3 Khung nhìn TOY-DEPT	48
Bảng 2.4 Bảng EMPLOYEE được chỉnh sửa.....	49
Bảng 2.5 Khung nhìn TOY-DEPT được tự động chỉnh sửa	49
Bảng 3.1 Các kiến trúc mẫu thử DBMS và các sản phẩm thương mại	79
Bảng 3.2 Các kiến trúc bộ nhớ bắt buộc có trong SGA	103
Bảng 3.2 Các kiến trúc bộ nhớ tùy chọn trong SGA	103
Bảng 3.3 Một số tiến trình nền	105
Bảng 3.4 Tuân thủ ACID của MySQL	109
Bảng 3.5 Quản lý lưu trữ trong MySQL.....	110
Bảng 3.6. Các bản vá lỗi của Oracle.....	118
Bảng 3.7: Các thủ tục của gói DBMS_FGA	147
Bảng 4.1 Các cảnh báo bảo mật cho ứng dụng Oracle.....	203
Bảng 4.2 Cổng cho máy chủ ứng dụng Oracle	204
Bảng 5.1 Ví dụ về SDB dạng quan hệ	208
Bảng 5.2 Ví dụ về thống kê vĩ mô về công nhân.....	208
Bảng 5.3. Ví dụ SDB công nhân.....	217
Bảng 5.4. Ví dụ về X(C) của SDB công nhân	217
Bảng 5.5. Ví dụ SDB về các vụ tai nạn ô tô	236
Bảng 5.6. Một ví dụ khác về SDB công nhân	240
Bảng 5.7. SDB vĩ mô về công nhân.....	250
Bảng 5.8. SDB vĩ mô về công nhân được giấu ô	250
Bảng 5.9. SDB vĩ mô về công nhân được giấu bổ sung	251
Bảng 5.10 So sánh tiêu chuẩn dành cho các kỹ thuật dựa vào hạn chế.....	278
Bảng 5.11 So sánh tiêu chuẩn dành cho các kỹ thuật dựa vào gây nhiễu	280

DANH MỤC HÌNH VẼ

Hình 1.1. Vai trò quan trọng của cơ sở dữ liệu	4
Hình 1.2. Sơ đồ lược giản về hệ thống cơ sở dữ liệu	5
Hình 1.3. Hình ảnh về tạo lớp vỏ bọc của DBMS	6
Hình 1.4. Thể hiện dữ liệu và siêu dữ liệu.....	8
Hình 1.5. Lược đồ CSDL xây dựng theo hướng phân tích thiết kế.....	9
Hình 1.6. Các mức mô tả dữ liệu.....	11
Hình 1.7. Các nguy cơ đến với cơ sở dữ liệu	15
Hình 2.1 Thiết kế các quy tắc trao quyền	32
Hình 2.2 Kiểm soát truy nhập tùy ý.....	37
Hình 2.3 Thu hồi quyền đê quy	40
Hình 2.4 Một biến thể của thu hồi quyền đê quy	41
Hình 2.5 Thu hồi quyền không đê quy (ví dụ 1)	42
Hình 2.6 Thu hồi quyền không đê quy (ví dụ 2)	42
Hình 2.7 Mối quan hệ giữa các mô hình RBAC	45
Hình 2.8 RBAC (ví dụ 1).....	46
Hình 2.9 RBAC (ví dụ 2).....	46
Hình 2.10 Tấn công Trojan Horse đối với DAC	54
Hình 2.11 Luồng thông tin trong MAC	56
Hình 2.12 Cấu trúc lưới của các nhãn an toàn.....	58
Hình 2.13 Các lớp phân cấp của nhãn an toàn	58
Hình 2.14 Kiểm soát truy nhập bắt buộc	61
Hình 2.15 Mô hình Bell – Lapadula chống được tấn công Trojan Horse	62
Hình 2.16 Các thể hiện của quan hệ MLS “Project”	64
Hình 3.1 Kiến trúc chủ thể tin cậy.....	80
Hình 3.2 Các kiến trúc Woods Hole	82
Hình 3.3 Khoá toàn vẹn	83
Hình 3.4 Ví dụ tổng kiểm tra mật mã	84
Hình 3.5 Kiến trúc Integrity Lock	86

Hình 3.6 Bộ lọc	87
Hình 3.7 Giải pháp bộ lọc thay thế	90
Hình 3.8 Giải pháp khung nhìn cho phép tối đa	91
Hình 3.9 Kiến trúc Kernelized.....	92
Hình 3.10 Kiến trúc Replicated	94
Hình 3.11. Tổng quan các kiến trúc DBMS nhiều mức như sau.....	96
Hình 3.12 Oracle Instance và Database	99
Hình 3.13 Kiến trúc MySQL	105
Hình 3.14 Kiến trúc Microsoft SQL Server	111
Hình 3.15. Cơ sở dữ liệu riêng ảo.....	138
Hình 3.16 Kiến trúc OLS	142
Hình 3.17: Ví dụ về chính sách kiểm toán	147
Hình 3.18 Ví dụ về thủ tục DBMS_FGA.ADD_POLICY	148
Hình 3.19. Mã hóa dữ liệu trong suốt trong Oracle.....	150
Hình 4.1 Mô hình ứng dụng cơ sở dữ liệu.....	152
Hình 4.2 liệt kê danh sách các tài khoản đang dùng để truy nhập CSDL và chương trình được sử dụng	160
Hình 4.3 Sử dụng một tường lửa SQL giữa ứng dụng và cơ sở dữ liệu.....	163
Hình 4.4 Form đăng nhập	180
Hình 4.5 Thông tin đăng nhập được đính vào danh sách chuyến bay (sau một câu lệnh tiêm UNION)	185
Hình 4.6 Lấy được một danh sách tất cả các đối tượng người dùng bằng cách sử dụng một tấn công UNION.....	186
Hình 4.7 Thêm một tin nhắn vào một bảng tin.....	187
Hình 4.8 Các tin nhắn trên bảng tin	187
Hình 4.9 Áp dụng việc thực hành tốt nhất cơ chế đặc quyền tối thiểu để hạn chế kết quả hợp pháp từ những điểm yếu ứng dụng (Before – top; After – bottom)	192
Hình 4.10 Báo cáo chi tiết về truy nhập ứng dụng – ai, cái gì, như thế nào	193
Hình 4.11 Các chính sách để cảnh báo và chặn các tấn công SQL injection	198
Hình 5.1. Website cơ sở dữ liệu thông kê của WTO	211

Hình 5.2. Tần công suy diễn thống kê	212
Hình 5.3. Làm lộ SDB và kiến thức bổ sung	216
Hình 5.4. Kỹ thuật dựa vào hạn chế	221
Hình 5.5. Kỹ thuật gây nhiễu dữ liệu	222
Hình 5.6. Kỹ thuật gây nhiễu đầu ra	222
Hình 5.7. Thống kê vĩ mô về công nhân theo năm sinh, giới tính và phòng	223
Hình 5.8 Lưới bảng trên các thuộc tính NamSinh, GioiTinh, MaPhong	225
Hình 5.9. Các bảng 2-chiều	225
Hình 5.10. Các bảng 1-chiều	226
Hình 5.11. Mô hình lưới 2	226
Hình 5.12 Các bảng ở các mức gộp khác nhau	228
Hình 5.13. Mô hình khái niệm áp dụng cho Employee SDB	231
Hình 5.14. Kỹ thuật kiểm soát kích cỡ tập truy vấn	232
Hình 5.15. Kỹ thuật gộp	246
Hình 5.16. Ví dụ về kỹ thuật gộp	248
Hình 5.17 Ví dụ về cây phân hoạch	253
Hình 5.18. Mô tả kỹ thuật gây nhiễu dữ liệu	257
Hình 5.19. Mô tả kỹ thuật gây nhiễu đầu ra	271

MỞ ĐẦU

Trong một xã hội hiện đại, cơ sở dữ liệu đóng một vai trò hết sức quan trọng. Chúng ta có thể thấy cơ sở dữ liệu tham gia vào hầu hết các lĩnh vực, sự ngưng trệ hay hoạt động thiếu chính xác của nó có thể gây ra các hậu quả khó lường. Đặc biệt, khi xã hội chuyển sang giai đoạn xã hội hoá thông tin và nền kinh tế chuyển sang nền kinh tế số hoá với mức độ liên kết chặt chẽ, với quy mô toàn cầu thì vai trò của cơ sở dữ liệu càng trở nên quan trọng. Do vậy, các vấn đề tin cậy, toàn vẹn và bí mật thông tin trong các cơ sở dữ liệu là những vấn đề cần được đầu tư nghiên cứu dưới cả góc độ lý thuyết và triển khai thực tiễn.

Bảo vệ thông tin trong các cơ sở dữ liệu là một vấn đề không đơn giản. Hiện nay đã có một số mô hình an toàn cơ sở dữ liệu được nghiên cứu nhiều về mặt lý thuyết và có các triển khai ứng dụng nhất định. Song để có thể triển khai được các mô hình này, chúng ta cần phải xây dựng cùng với các hệ quản trị cơ sở dữ liệu (DBMS), hoặc ít ra là cũng xây trên nền một Engine cơ sở dữ liệu quan hệ nào đó.

Tài liệu này sẽ cung cấp một số kiến thức cơ bản liên quan đến việc đảm bảo an toàn cho các cơ sở dữ liệu. Trong đó cũng cung cấp một số mô hình bảo vệ cơ sở dữ liệu được cài đặt trên một số hệ quản trị cơ sở dữ liệu quen thuộc để bạn đọc có thể hình dung được dễ dàng hơn và hiểu rõ hơn về những kỹ thuật an toàn được thể hiện như thế nào trong các hệ quản trị cơ sở dữ liệu. Chương 1 giới thiệu những khái niệm cơ bản, cung cấp một số hiểm họa, tấn công vào cơ sở dữ liệu và một số yêu cầu bảo vệ cơ sở dữ liệu. Chương 2 giới thiệu một số mô hình và chính sách an toàn cơ sở dữ liệu. Chương 3 liên quan đến vấn đề thiết kế, các vấn đề an toàn mức DBMS, đồng thời giới thiệu các cơ chế an toàn của hệ quản trị cơ sở dữ liệu Oracle, một hệ quản trị mạnh hiện nay. Chương 4 mô tả các cơ chế an toàn mức ứng dụng cơ sở dữ liệu. Chương 5, đi sâu vào cơ sở dữ liệu thống kê và vấn đề tách công suy diễn thống kê.

Khi biên soạn tài liệu này, chúng tôi cũng giả định rằng các độc giả đã có các kiến thức cơ bản về cơ sở dữ liệu, cũng như mật mã.

CHƯƠNG 1. TỔNG QUAN VỀ AN TOÀN

CƠ SỞ DỮ LIỆU

1.1 GIỚI THIỆU

Sự phát triển lớn mạnh của công nghệ thông tin trong những năm qua đã dẫn đến việc sử dụng rộng rãi hệ thống máy tính trong mọi tổ chức cá nhân và công cộng, chẳng hạn như: ngân hàng, trường học, tổ chức dịch vụ và sản xuất, bệnh viện, thư viện, quản lý phân tán và tập trung. Độ tin cậy của phần cứng, phần mềm ngày một được nâng cao cùng với việc liên tục giảm giá, tăng kỹ năng chuyên môn của các chuyên viên thông tin và sự sẵn sàng của các công cụ trợ giúp đã góp phần khuyến khích việc sử dụng dịch vụ máy tính một cách rộng rãi. Vì vậy, dữ liệu được lưu giữ và quản lý trong các hệ thống máy tính nhiều hơn. Các hệ quản trị cơ sở dữ liệu (DBMS) đã đáp ứng được các yêu cầu về lưu giữ và quản lý dữ liệu.

Nhiều phương pháp luận thiết kế cơ sở dữ liệu đã được phát triển nhằm hỗ trợ các yêu cầu thông tin khác nhau gắn với môi trường làm việc của ứng dụng. Các mô hình dữ liệu khái niệm và logic, cùng với những ngôn ngữ thích hợp, các công cụ định nghĩa dữ liệu, thao tác và hỏi đáp dữ liệu cũng đã được nghiên cứu. Mục tiêu là đưa ra các DBMS có khả năng quản trị và khai thác dữ liệu tốt.

Một đặc điểm cơ bản của DBMS là khả năng quản lý đồng thời nhiều giao diện ứng dụng. Mỗi ứng dụng có một cái nhìn thuần nhất về cơ sở dữ liệu, có nghĩa là có cảm giác chỉ mình nó đang khai thác cơ sở dữ liệu. Đây là một yêu cầu hết sức quan trọng đối với các DBMS, ví dụ cơ sở dữ liệu của ngân hàng với các khách hàng trực tuyến của nó; hoặc cơ sở dữ liệu của các hãng hàng không với việc đặt vé trước.

Xử lý phân tán đã góp phần phát triển và tự động hóa các hệ thống thông tin. Ngày nay, đơn vị xử lý thông tin của các tổ chức và các chi nhánh ở xa của nó có thể giao tiếp với nhau một cách nhanh chóng thông qua các mạng

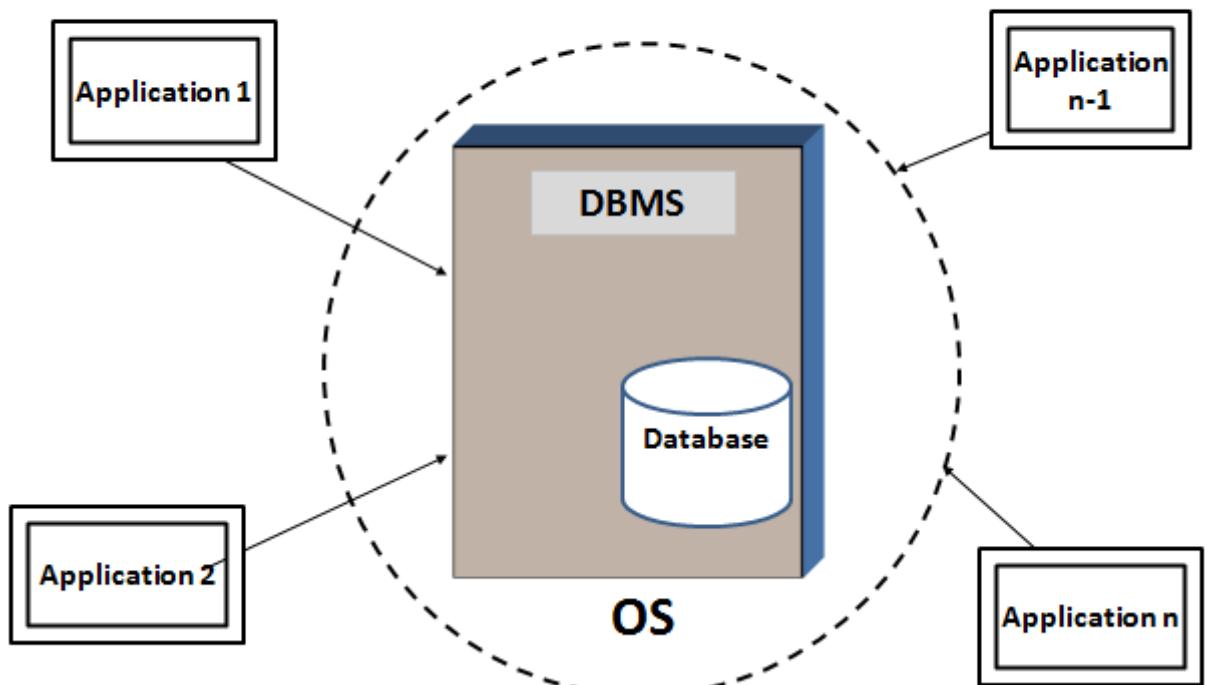
máy tính, vì vậy cho phép truyền tải rất nhanh các khối dữ liệu lớn. Việc sử dụng rộng rãi các cơ sở dữ liệu phân tán và tập trung đã đặt ra nhiều yêu cầu nhằm đảm bảo các chức năng thương mại và an toàn dữ liệu.

Trong thực tế, các sự cố trong môi trường cơ sở dữ liệu không chỉ ảnh hưởng đến từng người sử dụng hoặc ứng dụng, mà còn ảnh hưởng tới toàn bộ hệ thống thông tin. Thực tế cho thấy, sự cố về an ninh xảy ra với cơ sở dữ liệu có thể ảnh hưởng nghiêm trọng đến danh tiếng của công ty và quan hệ với khách hàng. Trong khi đó, đa số các công ty hiện nay tập trung nguồn lực vào bảo vệ dữ liệu trên đường truyền, mà vấn đề bảo vệ dữ liệu nằm trong cơ sở dữ liệu chưa được quan tâm đúng mức.

Thông tin luôn là một tài sản vô giá của các tổ chức, doanh nghiệp hay cá nhân và họ thường sử dụng các DBMS để lưu trữ, quản lý tập trung tất cả các thông tin quý giá của mình. Hiển nhiên hệ thống này sẽ là tiêu điểm tấn công của những người có mục đích xấu. Ở mức độ nhẹ, các tấn công sẽ làm hệ thống cơ sở dữ liệu bị hỏng hóc. Nghiêm trọng hơn, các thông tin sống còn của tổ chức, doanh nghiệp có thể bị tiết lộ (như chiến lược kinh doanh, các thông tin về khách hàng, nhà cung cấp, tài chính, mức lương nhân viên,...) và được đem bán cho các doanh nghiệp đối thủ. Ngoài ra các thông tin nhạy cảm, riêng tư của một cá nhân nào đó có thể bị phơi bày. Tất cả những điều đó có thể làm ảnh hưởng đến danh tiếng, uy tín, thậm chí làm các tổ chức, doanh nghiệp trở nên điêu đứng. Ngoài ra còn rất nhiều các hậu quả khôn lường ảnh hưởng đến các tổ chức, doanh nghiệp hay cá nhân, những người sở hữu cơ sở dữ liệu.

Cơ sở dữ liệu có một vai trò vô cùng quan trọng, nó được ví như trái tim của một ứng dụng. Khi cơ sở dữ liệu gặp sự cố có thể làm cho ứng dụng cơ sở dữ liệu bị ngừng trệ, ảnh hưởng rất nhiều đến uy tín, chất lượng và tiền bạc. Như vậy, có thể nói cơ sở dữ liệu là trung tâm của mọi vấn đề, là một trọng tâm mà các công ty, doanh nghiệp, tổ chức, cá nhân cần dành nhiều thời gian, công sức, nhân lực để bảo vệ. Từ đó nhất thiết phải có các biện pháp, cơ chế, chính sách để bảo vệ cơ sở dữ liệu cho mọi tình huống.

Database = the heart



Hình 1.1. Vai trò quan trọng của cơ sở dữ liệu

An toàn thông tin trong cơ sở dữ liệu

An toàn thông tin trong cơ sở dữ liệu bao gồm 3 yếu tố chính: *tính bí mật*, *toàn vẹn* và *sẵn sàng*.

Đảm bảo *tính bí mật* (*Confidentiality*): có nghĩa là ngăn chặn/phát hiện/cản trở những truy nhập thông tin trái phép. Nói chung, tính bí mật được sử dụng để bảo vệ dữ liệu trong những môi trường bảo mật cao như các trung tâm quân sự hay kinh tế quan trọng. Bảo vệ tính riêng tư của dữ liệu.

Đảm bảo *tính toàn vẹn* (*Integrity*): có nghĩa là ngăn chặn/phát hiện/cản trở các sửa đổi thông tin trái phép.

Đảm bảo *tính sẵn sàng* (*Availability*): có nghĩa là ngăn chặn/phát hiện/cản trở sự từ chối trái phép các truy nhập hợp pháp đến dịch vụ trong hệ thống.

1.2 MỘT SỐ KHÁI NIỆM CƠ BẢN

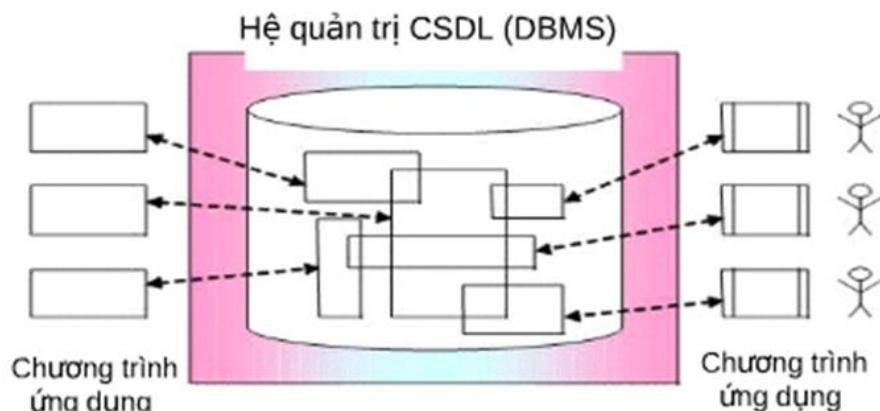
Cơ sở dữ liệu là một tập hợp dữ liệu không nhất thiết đồng nhất, có quan hệ với nhau về mặt lôgic và được phân bố trên một mạng máy tính.

Hệ thống phần mềm cho phép quản lý, thao tác trên cơ sở dữ liệu, tạo ra sự trong suốt phân tán với người dùng gọi là hệ quản trị cơ sở dữ liệu (DBMS).

Một số DBMS được sử dụng phổ biến hiện nay như: MS Access, MS SQL Server, MySQL, Oracle, DB2, Sybase, Informix, v.v...

1.2.1 Hệ cơ sở dữ liệu

Một hệ cơ sở dữ liệu bao gồm 4 thành phần: dữ liệu, phần cứng, phần mềm và người dùng.



Hình 1.2. Sơ đồ lược giản về hệ thống cơ sở dữ liệu

1.2.1.1 Dữ liệu

Trong cơ sở dữ liệu, dữ liệu có hai đặc trưng chính là:

- *Tính tích hợp*: Cơ sở dữ liệu là nơi tập hợp nhiều hồ sơ, và người ta cố gắng loại bỏ đến mức tối đa các dữ liệu dư thừa. Lấy ví dụ, trong một cơ quan có danh sách nhân sự và bảng lương hàng tháng. Trong danh sách nhân sự, chúng ta có toàn bộ họ tên của cán bộ nhân viên. Trong bảng lương hàng tháng, chúng ta cũng thấy bên cạnh các cột thông tin khác, là cột họ tên cán bộ nhân viên. Rõ ràng, trong cơ sở dữ liệu, không cần thiết phải lưu danh sách

họ tên cán bộ nhân viên một lần nữa, mà trường thông tin này có thể lấy ra nhờ sử dụng tham chiếu.

- *Tính chia sẻ*: Cơ sở dữ liệu là nơi cho phép nhiều người dùng truy nhập – tùy theo mục đích sử dụng của từng người. Hơn nữa, nhiều người dùng có thể truy nhập đồng thời (chứ không phải người này phải đợi người kia truy nhập xong mới đến lượt).

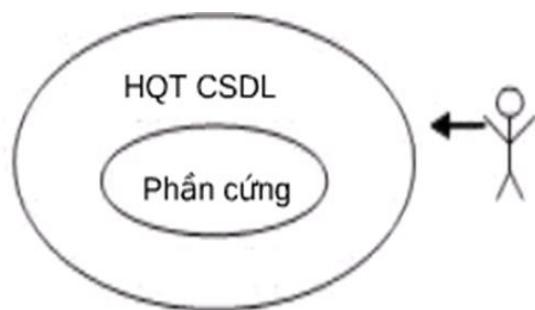
1.2.1.2. Phần cứng

Phần cứng của hệ thống cơ sở dữ liệu bao gồm:

- Bộ nhớ ngoài: chủ yếu là đĩa từ, cụ thể hơn là đĩa cứng cùng với các bộ phận điều khiển khác như khôi vào/ra ổ đĩa, khôi điều khiển,... được sử dụng để lưu trữ dữ liệu
- Bộ xử lý và bộ nhớ trong: dùng để chạy các phần mềm của hệ thống cơ sở dữ liệu. Khối kết nối gồm card mạng/modem được sử dụng để kết nối hệ thống cơ sở dữ liệu với thế giới bên ngoài.

1.2.1.3 Phần mềm

Đứng trung gian giữa phần vật lý (là nơi thực sự lưu trữ dữ liệu) và người dùng là khái niệm phần mềm quản trị cơ sở dữ liệu, được gọi là hệ quản trị cơ sở dữ liệu hay DBMS. Tất cả các phép toán thực hiện trên cơ sở dữ liệu đều phải thông qua DBMS như: thêm mới, hủy bỏ, trích xuất, cập nhật dữ liệu,... DBMS tạo ra sự tương tác đối với người dùng. Người dùng chỉ nhìn thấy cơ sở dữ liệu thông qua các công cụ mà DBMS cung cấp, chứ không cần quan tâm đến phần vật lý của cơ sở dữ liệu.



Hình 1.3. Hình ảnh về tạo lớp vỏ bọc của DBMS

1.2.1.4 Người dùng

Người dùng cơ sở dữ liệu có thể chia thành các lớp như sau:

Lớp thứ nhất: Lập trình viên cơ sở dữ liệu, là người viết chương trình ứng dụng sử dụng cơ sở dữ liệu thông qua một ngôn ngữ nào đó, như: COBOL, C++, C#, ASP, PHP,... Các chương trình này sử dụng các phép toán cơ sở dữ liệu thông thường như: thêm, sửa, xóa, ... chủ yếu sử dụng các câu lệnh SQL. Các chương trình có thể được viết theo lô các lệnh hoặc cũng có thể hoạt động trực tuyến – nghĩa là giao tiếp trực tiếp với DBMS. Chức năng hoạt động trực tuyến thường được sử dụng để quản trị cơ sở dữ liệu.

Lớp thứ hai: người dùng cuối, là người sử dụng các chương trình đã lập sẵn để giao tiếp với cơ sở dữ liệu. Các chương trình đã lập sẵn gồm các chương trình được lập bởi các lập trình viên hoặc là một phần của DBMS. Phần lớn các DBMS đều cung cấp nhiều tiện ích lập sẵn. Một trong các tiện ích cơ bản đó là giao diện truy vấn. Trong giao diện này, người dùng có thể đưa ra các câu lệnh SQL và phần mềm sẽ cho kết quả của câu lệnh đó.

Lớp thứ ba: Những người quản trị cơ sở dữ liệu (DBA), là người làm công tác quản trị cơ sở dữ liệu.

Đối với một hệ thống quản lý an toàn cơ sở dữ liệu, chúng ta cần có một số lớp người dùng sau:

Người quản lý ứng dụng: có trách nhiệm đối với việc phát triển và duy trì, hoặc các chương trình thư viện.

DBA: quản lý các lược đồ khái niệm và lược đồ bên trong của cơ sở dữ liệu.

Nhân viên an toàn: xác định các quyền truy nhập, các tiên đề, thông qua các quy tắc trong một ngôn ngữ thích hợp (có thể là DDL, hoặc DML)

Kiểm toán viên: chịu trách nhiệm kiểm tra các yêu cầu kết nối và các câu hỏi truy nhập, nhằm phát hiện ra các xâm phạm quyền.

...

Ngoài các khái niệm trên, trong cơ sở dữ liệu chúng ta lưu ý hai khái niệm cơ bản là: thể hiện (Instance) và lược đồ (Schema).

1.2.1.5 Thể hiện

Một khi cơ sở dữ liệu đã được thiết kế, thường người ta quan tâm tới “bộ khung” hay còn gọi là “mẫu” của cơ sở dữ liệu. Dữ liệu hiện có trong cơ sở dữ liệu gọi là “thể hiện” của cơ sở dữ liệu. Chẳng hạn ở ví dụ dưới đây, ta thấy được sự khác biệt giữa thể hiện của cơ sở dữ liệu và siêu dữ liệu.

Data instance					metadata				
EmpNo	EmpName	DeptNo	Salary	Birthdate	Column Name	Data Type	Length	Allow Nulls	
1	Lan	203	2000	03/28/1970	EmpNo	int	4		
2	Minh	104	3000	12/11/1982	EmpName	varchar	50		
3	Huyền	300	3500	10/30/1983	Deptno	int	4	✓	
					salary	decimal	9	✓	
					birthdate	datetime	8	✓	

Hình 1.4. Thể hiện dữ liệu và siêu dữ liệu

1.2.1.6 Lược đồ

Thường “bộ khung” nêu trên bao gồm một số danh mục, hoặc chỉ tiêu hay một số kiểu của thực thể trong cơ sở dữ liệu. Giữa các thực thể có thể có mối quan hệ nào đó với nhau, ở đây sử dụng thuật ngữ “lược đồ” để thay cho khái niệm “bộ khung”.

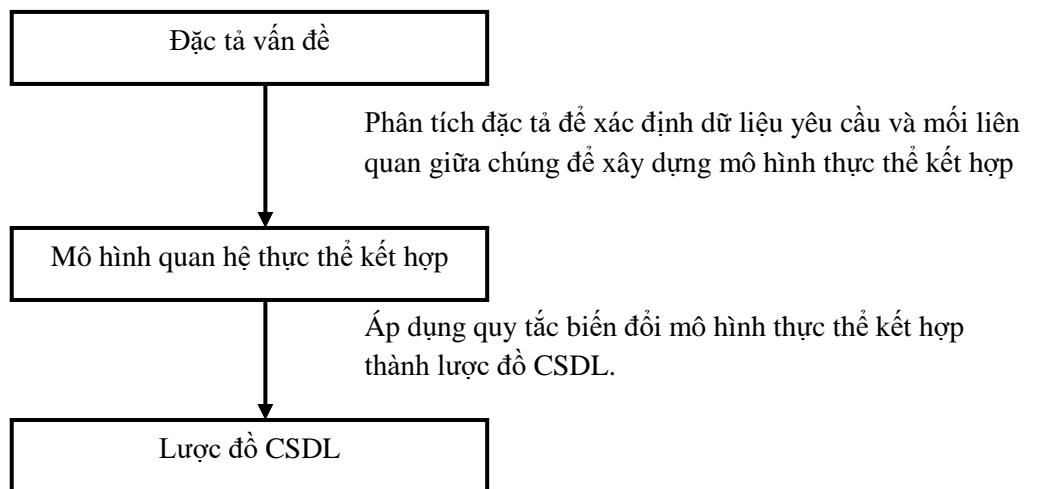
1.2.2 Thiết kế cơ sở dữ liệu

Trong thiết kế cơ sở dữ liệu, chúng ta cần phân biệt *pha khái niệm* và *pha logic*. Các mô hình khái niệm và logic tương ứng thường dùng để mô tả cấu trúc của cơ sở dữ liệu. Trong đó:

- **Mô hình logic:** phụ thuộc vào DBMS (ví dụ mô hình quan hệ, mô hình phân cấp, mô hình mạng hay một mô hình mới đang phát triển là mô hình hướng đối tượng).
- **Mô hình khái niệm** độc lập với DBMS. Mô hình quan hệ thực thể E - R (Entity Relationships) là một trong các mô hình khái niệm phổ biến

nhất, được xây dựng dựa trên khái niệm thực thể và mối quan hệ giữa các thực thể. *Thực thể* được xem như là lớp các đối tượng của thế giới hiện thực được mô tả bên trong cơ sở dữ liệu và *quan hệ* mô tả mối liên hệ giữa các lớp đối tượng đó.

Trong quá trình thiết kế logic, lược đồ khái niệm được chuyển sang lược đồ logic. Ta có thể xây dựng lược đồ logic cơ sở dữ liệu theo các mức như hình vẽ sau:



Hình 1.5. Lược đồ CSDL xây dựng theo hướng phân tích thiết kế

Các ngôn ngữ sẵn có trong DBMS bao gồm: ngôn ngữ định nghĩa dữ liệu (DDL), ngôn ngữ thao tác dữ liệu (DML) và ngôn ngữ truy vấn (QL).

DDL (Data Definition Language) là ngôn ngữ máy tính để định nghĩa lược đồ CSDL logic. Ví dụ về ngôn ngữ DDL như: lược đồ XML (Extensible Markup Language), hay một tập các câu lệnh của SQL (Structured Query Language) như: CREATE, DROP, ALTER, vv...

Ví dụ, lệnh **Create** sau được viết trong Oracle sẽ tạo ra một bảng có tên là *Employees*

```

CREATE TABLE Employees(
    EmpID INT NOT NULL,
  
```

```

    Name varchar(30) NOT NULL,
    Address varchar(30) NOT NULL,
    Salary INT NOT NULL,
    Email varchar(40) NOT NULL,
    Phone varchar(11) NOT NULL
);

```

DML(Data Manipulation Language) là họ các ngôn ngữ máy tính được người dùng sử dụng để tìm kiếm, chèn, xóa và cập nhật dữ liệu trong một CSDL. Các câu lệnh DML như các câu lệnh của SQL: SELECT, INSERT, UPDATE, DELETE. Ngôn ngữ DML có thể nhúng trong một ngôn ngữ lập trình thông thường, gọi là ngôn ngữ nhúng. Vì vậy, các ứng dụng sử dụng ngôn ngữ lập trình có thể đưa vào các câu lệnh của DML cho các phép toán hướng dữ liệu.

Ví dụ một số câu lệnh DML được viết trong Oracle dưới đây:

- **Select**

```

SELECT EmpID, Name
FROM Employees WHERE (EmpID = 10);

```

- **Insert**

```

INSERT INTO Employees
VALUES (101, 'Trang', 'Ha noi', 2000,'trang@yahoo.com',
'0985512621');

```

- **Update**

```

UPDATE Employees SET Name = 'Minh'
WHERE EmpID = 101;

```

- **Delete**

```

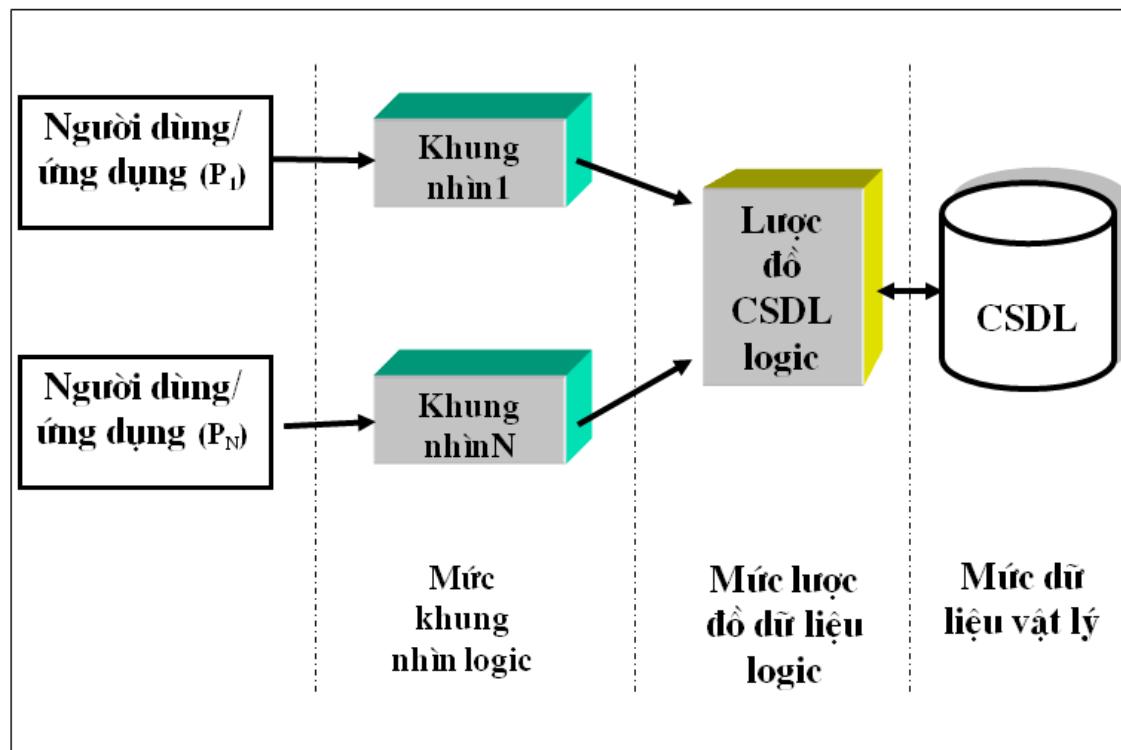
DELETE FROM Employees
WHERE EmpID = 101

```

QL (Query Language) là ngôn ngữ khai báo hỗ trợ cho những người dùng cuối, giúp người dùng tìm kiếm dữ liệu trong cơ sở dữ liệu. Ví dụ về QL như là câu lệnh lựa chọn trong SQL là: SELECT.

1.2.3 Các mức mô tả dữ liệu

DBMS mô tả dữ liệu theo nhiều mức khác nhau. Mỗi mức cung cấp một mức trừu tượng về cơ sở dữ liệu. Trong DBMS có thể có các mức mô tả sau:



Hình 1.6. Các mức mô tả dữ liệu

Khung nhìn logic

Việc xây dựng các khung nhìn tuỳ thuộc các yêu cầu của mô hình logic và các mục đích của ứng dụng. Khung nhìn logic mô tả một phần lược đồ cơ sở dữ liệu logic. Nói chung, người ta thường sử dụng DDL để định nghĩa các khung nhìn logic, DML để thao tác trên các khung nhìn này.

Lược đồ dữ liệu logic

Ở mức này, mọi dữ liệu trong cơ sở dữ liệu được mô tả bằng mô hình logic của DBMS. Các dữ liệu và quan hệ của chúng được mô tả thông qua DDL của DBMS. Các thao tác khác nhau trên lược đồ logic được xác định thông qua DML của DBMS đó.

Lược đồ dữ liệu vật lý

Mức này mô tả cấu trúc lưu trữ dữ liệu trong các file trên bộ nhớ ngoài. Dữ liệu được lưu trữ dưới dạng các bản ghi (có độ dài cố định hay thay đổi) và các con trỏ trỏ tới bản ghi.

1.2.4 Ngôn ngữ SQL

Mỗi DBMS đều phải có ngôn ngữ giao tiếp giữa người sử dụng với cơ sở dữ liệu. SQL (Structure Query Language) là ngôn ngữ hỏi đáp có cấu trúc cho phép người sử dụng khai thác CSDL để truy vấn các thông tin cần thiết trong CSDL.

Những năm 1975-1976, IBM lần đầu tiên đưa ra DBMS kiểu quan hệ mang tên SYSTEM-R với ngôn ngữ giao tiếp cơ sở dữ liệu là SEQUEL (Structured English Query Language). Năm 1976 ngôn ngữ SEQUEL được cải tiến thành SEQUEL-2, khoảng năm 1978-1979 SEQUEL-2 được cải tiến và đổi tên thành ngôn ngữ truy vấn có cấu trúc (Structured Query Language). Cuối năm 1979 hệ quản trị CSDL được cải tiến thành SYSTEM-R*. Năm 1986 viện tiêu chuẩn quốc gia Mỹ (American National Standards Institute – ANSI) đã công nhận và chuẩn hóa ngôn ngữ SQL và sau đó tổ chức tiêu chuẩn thế giới (International Standards Organization -ISO) cũng đã công nhận ngôn ngữ này. Đó là chuẩn SQL-86. tới nay SQL đã qua 3 lần chuẩn hóa (1989,1992,1996) để mở rộng các phép toán và tăng cường khả năng bảo mật và tính toàn vẹn dữ liệu.

1.3 CÁC HIỂM HỌA VÀ TẤN CÔNG ĐỐI VỚI CƠ SỞ DỮ LIỆU

Trước tiên, có thể lý giải tại sao cơ sở dữ liệu luôn nằm trong tầm ngắm của nhiều tin tặc. Bởi vì, cơ sở dữ liệu lưu giữ những thông tin quan trọng như: thông tin của của khách hàng, kế hoạch phát triển của một doanh nghiệp, các dự đoán kinh tế, và nhiều mục đích quan trọng khác... Cơ sở dữ liệu còn chứa nhiều thông tin riêng tư của các cá nhân mà tin tặc có thể thu được bằng cách truy vấn những dữ liệu công khai. Hơn nữa, sẽ có lợi cho một tin tặc khi tấn công vào cơ sở dữ liệu hơn là nghe nén giao tiếp trên mạng. Mặt khác, dữ liệu thường được mã hóa trên đường truyền nhưng lại lưu dưới dạng rõ trong cơ sở dữ liệu.

1.3.1 Hiểm họa ngẫu nhiên và có chủ ý

Một *hiểm họa* có thể được xác định khi đối phương (người, hoặc nhóm người) sử dụng các kỹ thuật đặc biệt để tiếp cận nhằm khám phá, sửa đổi trái phép thông tin quan trọng do hệ thống quản lý.

Các *hiểm họa* an toàn có thể được phân thành hai loại: *hiểm họa ngẫu nhiên* và *hiểm họa có chủ ý*.

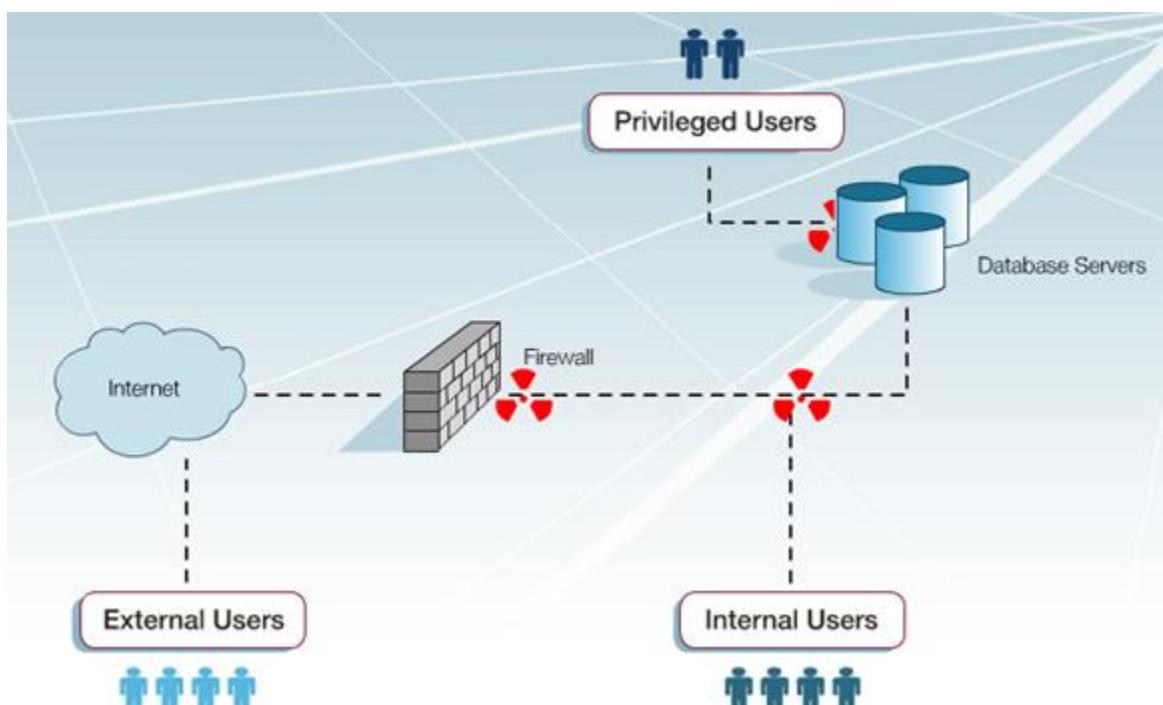
- **Hiểm họa ngẫu nhiên** thường liên quan tới các trường hợp sau:
 - ✓ Các thảm họa trong thiên nhiên, chẳng hạn như động đất, hoả hoạn, lụt lội... có thể phá hỏng các hệ thống phần cứng, hệ thống lưu giữ số liệu, ảnh hưởng đến tính toàn vẹn và sẵn sàng của hệ thống.
 - ✓ Các lỗi phần cứng hay phần mềm có thể dẫn đến việc áp dụng các chính sách an toàn không đúng, từ đó cho phép truy nhập, đọc, sửa đổi dữ liệu trái phép, hoặc từ chối dịch vụ đối với người dùng hợp pháp.
 - ✓ Các sai phạm vô ý do con người gây ra, chẳng hạn như nhập dữ liệu đầu vào không chính xác, hay sử dụng các ứng dụng không đúng, hậu quả cũng tương tự như các nguyên nhân do lỗi phần mềm hay lỗi kỹ thuật gây ra.
- **Hiểm họa có chủ ý** liên quan đến hai lớp người dùng sau:
 - ✓ Người dùng hợp pháp nhưng lạm dụng quyền, sử dụng vượt quá quyền hạn được phép của họ.
 - ✓ Người dùng truy nhập thông tin trái phép, có thể là những người nằm ngoài tổ chức hay bên trong tổ chức. Họ tiến hành các hành vi phá hoại phần mềm CSDL hay phần cứng của hệ thống, hoặc đọc ghi dữ liệu trái phép. Trong cả hai trường hợp trên, họ đều thực hiện với chủ ý rõ ràng.

1.3.2 Tấn công bên trong và bên ngoài

Với loại *hiểm họa* có chủ ý, chúng ta có thể coi đó là các tấn công đối với cơ sở dữ liệu. Những tấn công này có thể đến từ bên trong hoặc bên ngoài hệ thống. Với các tấn công bên ngoài, kẻ tấn công không phải nhân viên của tổ chức, doanh nghiệp, họ hầu như không biết về kiến trúc bên trong của hệ thống nên việc tấn công có vẻ khó khăn hơn. Để ngăn chặn các tấn công này,

một số hệ thống sử dụng hệ thống tường lửa, hệ thống phát hiện và ngăn chặn xâm nhập nhằm giới hạn khả năng truy xuất thông tin của những khách hàng hay người dùng chân chính. Tuy nhiên, các hệ thống này không chống được các tấn công ngày một tinh vi cũng như các lỗ hổng bảo mật xuất hiện ngày càng nhiều.

Các đe dọa còn đến từ bên trong, từ chính các nhân viên của tổ chức, doanh nghiệp - những người mặc nhiên được truy nhập vào hệ thống cơ sở dữ liệu và được mặc nhiên tin tưởng. Điều đặc biệt là hệ thống tường lửa hoàn toàn vô hiệu trước đối tượng này. Đáng nói hơn nữa là các biện pháp bảo vệ thông tin từ bên trong không ít thì nhiều bị các tổ chức, doanh nghiệp đầu tư chưa đúng mức hoặc hoàn toàn bị “quên lãng”. So với tấn công bên ngoài, các tấn công bên trong khó kiểm soát hơn vì những người dùng này đã biết kiến trúc bên trong của hệ thống và có những quyền hạn nhất định. Nguy hiểm hơn, bản thân những người dùng có đặc quyền như: quản trị viên, những người có đặc quyền cũng là một mối nguy cơ tiềm tàng cho sự an toàn của hệ thống cơ sở dữ liệu. Do đó, cần phải có những biện pháp toàn diện, tích cực, sử dụng cả biện pháp kỹ thuật và giáo dục người dùng mới có khả năng để bảo vệ cơ sở dữ liệu.



Hình 1.7. Các nguy cơ đến với cơ sở dữ liệu

1.3.3 Mười hiểm họa hàng đầu tới an toàn cơ sở dữ liệu

Cơ sở dữ liệu của các tổ chức, doanh nghiệp luôn là mục tiêu của nhiều tấn công, chính vì vậy để giúp các tổ chức, doanh nghiệp nhận ra và xử lý được những mối đe dọa, hiểm họa tới hệ thống cơ sở dữ liệu của mình, trung tâm phòng thủ ứng dụng (Application Defense Center) thuộc công ty Imperva đã liệt kê ra mười hiểm họa hàng đầu đối với sự an toàn của cơ sở dữ liệu.

Xin giới thiệu thêm về công ty Imperva, đó là một công ty hàng đầu về an toàn dữ liệu (đặt trụ sở tại Mỹ), chuyên cung cấp các giải pháp bảo vệ dữ liệu cho doanh nghiệp và ngăn chặn các hành vi lấy cắp dữ liệu nhạy cảm của hacker và của những người bên trong ác ý, bằng cách đảm bảo an toàn dữ liệu quan trọng ba lĩnh vực chính là: cơ sở dữ liệu, các hệ thống file và các ứng dụng web. Bên cạnh đó, Imperva có một Gateway đảm bảo an toàn cơ sở dữ liệu có tên là SecureSphere. Công an toàn cơ sở dữ liệu này có khả năng bảo vệ chống lại mười hiểm họa hàng đầu đối với an toàn cơ sở dữ liệu được liệt kê sau đây:

1. Lạm dụng quyền vượt mức (Excessive Privilege Abuse)
2. Lạm dụng quyền hợp pháp (Legitimate Privilege Abuse)
3. Nâng cấp quyền bất hợp pháp (Unauthorized Privilege Elevation)
4. Lợi dụng các điểm yếu nền tảng (Platform Vulnerabilities)
5. SQL Injection
6. Lợi dụng dấu vết kiểm toán yếu (Weak Audit Trail)
7. Từ chối dịch vụ (Denial of Service)
8. Lợi dụng các điểm yếu trong giao thức giao tiếp cơ sở dữ liệu (Database Communication Protocol Vulnerabilities)
9. Lợi dụng sự xác thực yếu (Weak Authentication)
10. Lợi dụng sự sơ hở của dữ liệu dự phòng (Backup Data Exposure)

Sau đây là một vài mô tả về 10 hiểm họa an toàn cơ sở dữ liệu nói trên, đồng thời đưa ra cách ngăn chặn cũng như khả năng của công SecureSphere chống lại các hiểm họa này.

1. Lạm dụng quyền vượt mức (Excessive Privilege Abuse)

Khi người dùng (hay ứng dụng) được gán các đặc quyền truy nhập cơ sở dữ liệu vượt quá các yêu cầu trong chức năng công việc của họ, thì những đặc quyền này có thể bị lạm dụng cho các mục đích xấu. Ví dụ, một người phụ trách cho cơ sở dữ liệu của trường đại học với công việc là thay đổi các thông tin liên lạc của sinh viên, người này có thể lạm dụng quyền của mình với quyền cập nhật cơ sở dữ liệu để sửa đổi điểm của sinh viên (trái phép).

Một lý do đơn giản là những quản trị viên cơ sở dữ liệu vì bận rộn với công việc quản trị của mình nên không có thời gian để định nghĩa và cập nhật cơ chế kiểm soát quyền truy nhập mịn cho mỗi người dùng. Kết quả là một số lượng lớn người dùng được gán các đặc quyền truy nhập mặc định vượt xa so với yêu cầu công việc của họ.

Ngăn chặn lạm dụng quyền vượt mức – Bằng kiểm soát truy nhập mức truy vấn

Giải pháp cho hiểm họa cơ sở dữ liệu này, đó là sử dụng cơ chế kiểm soát truy nhập mức truy vấn (Query-level). Cơ chế này sẽ hạn chế các đặc quyền cơ sở dữ liệu bằng những toán tử SQL (SELECT, UPDATE,...) và dữ liệu được yêu cầu một cách tối thiểu. Độ mịn trong kiểm soát truy nhập dữ liệu không chỉ ở mức bảng mà còn tới các hàng, các cột cụ thể trong bảng. Trong ví dụ trên, nếu sử dụng cơ chế kiểm soát truy nhập mức truy vấn mịn một cách đầy đủ thì người phụ trách cơ sở dữ liệu ác ý của trường đại học chỉ được phép cập nhật cột thông tin liên lạc của sinh viên, và nếu anh ta cố gắng sửa đổi cột điểm thì hệ thống sẽ phát ra cảnh báo. Kiểm soát truy nhập mức truy vấn không chỉ có lợi trong việc phát hiện việc lạm dụng quyền vượt mức mà còn giúp ngăn chặn hầu hết những hành động ác ý trong mười hiểm họa an toàn cơ sở dữ liệu vừa nêu.

Ví dụ: Grant Update on SinhVien to Smith

Grant Update on SinhVien(Contact) to Smith.

2. *Lạm dụng quyền hợp pháp (Legitimate Privilege Abuse)*

Người dùng lạm dụng các đặc quyền hợp pháp của mình để thực hiện những mục đích không hợp pháp. Hãy xét một nhân viên chăm sóc sức khỏe với giả thuyết người này có mục đích xấu. Nhân viên này có quyền xem các bản ghi liên quan đến chỉ một bệnh nhân trong cơ sở dữ liệu qua một ứng dụng Web. Tuy nhiên, thường cấu trúc của ứng dụng Web sẽ hạn chế không cho phép người dùng được xem nhiều bản ghi về lịch sử khám chữa bệnh liên quan đến nhiều bệnh nhân một cách đồng thời. Khi đó, nhân viên này có thể bỏ qua hạn chế của ứng dụng Web đó bằng cách kết nối tới cơ sở dữ liệu bằng một trình khách khác, chẳng hạn MS-Excel. Trong đó, anh ta sử dụng MS-Excel với vé ủy nhiệm đăng nhập hợp lệ của mình, nhờ vậy anh ta có thể lấy ra và lưu lại tất cả các bản ghi về các bệnh nhân trong cơ sở dữ liệu.

Ở đây, cần xét hai rủi ro. Thứ nhất là người nhân viên có mục đích xấu, người sẵn sàng bán các bản ghi thông tin bệnh nhân để lấy tiền. Thứ hai (phổ biến hơn) là người nhân viên cẩu thả, người này truy xuất và lưu trữ một lượng lớn thông tin bệnh nhân vào máy khách của họ với mục đích công việc hợp pháp. Tuy nhiên, do lơ đãng, nhân viên này có thể để lộ dữ liệu bệnh nhân ở một máy tính đầu cuối, và nó trở thành điểm yếu cho Trojan, hay những kẻ lấy cắp máy tính,...

Ngăn chặn lạm dụng quyền hợp pháp – Bằng cách hiểu được bối cảnh của truy nhập cơ sở dữ liệu:

Giải pháp cho hiểm họa này chính là kiểm soát truy nhập cơ sở dữ liệu cần áp dụng cho không chỉ các truy vấn cụ thể như được mô tả ở trên, mà còn áp dụng cho bối cảnh xung quanh truy nhập cơ sở dữ liệu. Bằng cách bắt buộc tuân thủ chính sách cho các ứng dụng khách, thời gian trong ngày, vị trí, ... thì có thể nhận dạng được những người dùng đang sử dụng các quyền truy nhập hợp pháp của mình với hành động mờ ám, đáng nghi.

Với hiểm họa này, SecureSphere sử dụng công nghệ Hồ sơ động (Dynamic Profiling) để thực hiện kiểm soát truy nhập dựa vào bối cảnh.

Ngoài thông tin truy vấn (như giải pháp với hiềm họa thứ nhất – Lạm dụng quyền quá mức), công nghệ Hồ sơ động tạo ra một mô hình của ngữ cảnh xung quanh các tương tác cơ sở dữ liệu thông thường. Thông tin về ngữ cảnh cụ thể được lưu trong profile bao gồm: thời gian trong ngày, địa chỉ IP nguồn, khôi dữ liệu được truy xuất, trình khách ứng dụng,...

Bất kỳ kết nối nào đến cơ sở dữ liệu mà ngữ cảnh của nó không phù hợp với thông tin được lưu trong hồ sơ của người dùng đó, thì công SecureSphere sẽ phát ra cảnh báo. Chẳng hạn, người nhân viên có mục đích xấu trong ví dụ trên sẽ bị phát hiện bởi SecureSphere không chỉ vì việc sử dụng một trình khách MS-Exel không chuẩn mà còn vì khôi dữ liệu mà người dùng này truy xuất cũng không hợp lệ đối với một phiên đơn lẻ. Trong trường hợp cụ thể này, sự sai lệch trong cấu trúc của truy vấn MS-Exel không chuẩn đó, cũng gây ra một sự xâm phạm mức truy vấn (như phần hiềm họa đầu tiên)

3. Nâng cấp quyền bất hợp pháp (Unauthorized Privilege Elevation)

Kẻ tấn công có thể dựa trên các điểm yếu trong phần mềm cơ sở dữ liệu để biến các đặc quyền truy nhập của một người dùng bình thường thành quyền truy nhập của một người quản trị. Những điểm yếu này có thể tìm thấy trong các thủ tục được lưu, trong các hàm được xây dựng bên trong, trong việc thực thi giao thức, thậm chí trong các câu lệnh SQL. Ví dụ, một nhà phát triển phần mềm ở một tổ chức tài chính có thể lợi dụng một hàm chứa điểm yếu để thu được đặc quyền quản trị cơ sở dữ liệu. Với đặc quyền quản trị này, nhà phát triển ác ý đó có thể tắt các cơ chế kiểm toán, tạo những tài khoản ma, hay chuyển tiền bất hợp pháp, ...

Ngăn chặn nâng quyền bất hợp pháp – Bằng IPS và kiểm soát truy nhập mức truy vấn

Hiềm họa này có thể bị ngăn chặn bằng việc kết hợp các hệ thống ngăn chặn xâm nhập (IPS) và kiểm soát truy nhập mức truy vấn. IPS sẽ kiểm tra lưu lượng cơ sở dữ liệu để nhận ra các mẫu phù hợp với những điểm yếu đã biết. Chẳng hạn, với một hàm có điểm yếu đã biết, thì một IPS có thể chặn tất cả các truy nhập tới hàm đó, hoặc (nếu có thể) chỉ chặn những truy nhập (của các tấn công) tới hàm này.

Tuy nhiên, thật khó khăn để xác định một cách chính xác những yêu cầu truy nhập nào gắn với các tấn công khi sử dụng một mình IPS. Bởi vì nhiều truy nhập hợp pháp vẫn sử dụng các hàm cơ sở dữ liệu chứa điểm yếu. Do đó, IPS không có tùy chọn để chặn tất cả các truy nhập đến các hàm chứa điểm yếu. IPS phải phân biệt chính xác các hàm được sử dụng hợp pháp và các hàm được sử dụng cho một tấn công nào đó. Trong nhiều trường hợp, điều này là không thể do sự đa dạng của các tấn công, cho nên với những trường hợp như vậy, hệ thống IPS chỉ đưa ra cảnh báo (không chặn) đối với những truy nhập vào các hàm cơ sở dữ liệu chứa điểm yếu.

Để cải thiện độ chính xác, IPS có thể được gắn với một bộ chỉ thị tấn công khác, chẳng hạn như kiểm soát truy nhập truy vấn. Kiểm soát truy nhập truy vấn kiểm tra xem yêu cầu truy nhập có phù hợp với hoạt động của người dùng thông thường hay không, trong khi đó IPS sẽ kiểm tra xem yêu cầu cơ sở dữ liệu này có truy nhập vào một hàm chứa điểm yếu hay không. Nếu một yêu cầu cơ sở dữ liệu có truy nhập vào một hàm chứa điểm yếu và có hoạt động bất thường thì chắc chắn có một tấn công vào cơ sở dữ liệu đang hoạt động.

Với hiểm họa này, SecureSphere sử dụng kết hợp IPS và Hồ sơ động bằng cách tích hợp IPS với hồ sơ động (kiểm soát truy nhập truy vấn). Hai công nghệ này hoạt động cùng nhau sẽ đem lại khả năng bảo vệ chống lại nâng quyền bất hợp pháp với độ chính xác cao.

Để minh họa việc SecureSphere tích hợp IPS và Hồ sơ động, ta trở lại ví dụ với nhà phát triển phần mềm cho tổ chức tài chính ở ví dụ trước. Giả sử rằng, người này đang cố gắng lợi dụng lỗi tràn bộ đệm đã biết trong một hàm cơ sở dữ liệu để chèn mã độc nhằm nâng cấp các đặc quyền của anh ta tới quyền của nhà quản trị cơ sở dữ liệu. Trong trường hợp này, SecureSphere sẽ nhận ra được hai xâm phạm một cách đồng thời. Đầu tiên là khi một truy vấn bất kỳ đang cố gắng truy nhập vào một hàm chứa điểm yếu đã biết thì IPS sẽ phát hiện ra sự xâm nhập này. Thứ hai, với một truy vấn khác thường thì Hồ sơ động cũng có thể phát hiện ra đây là một xâm nhập. Bằng cách kết hợp hai xâm nhập này trong một yêu cầu truy nhập cơ sở dữ liệu từ cùng một người dùng, hệ thống có thể xác định một cách chính xác đây là một tấn công vào cơ

sở dữ liệu, đồng thời đưa ra một cảnh báo có độ ưu tiên cao hoặc đưa ra hành động ngăn chặn.

4. Lợi dụng các điểm yếu nền tảng (Platform Vulnerabilities)

Các điểm yếu trong hệ điều hành bên dưới (Windows 2000, UNIX, ...) và các điểm yếu trong các dịch vụ được cài đặt trên một máy chủ cơ sở dữ liệu có thể dẫn tới truy nhập bất hợp pháp, sự sửa đổi dữ liệu hay từ chối dịch vụ. Chẳng hạn, sâu Blaster lợi dụng một điểm yếu của Windows 2000 để tạo ra các điều kiện cho từ chối dịch vụ.

Ngăn chặn các tấn công nền tảng – Băng cập nhật phần mềm và ngăn chặn xâm nhập

Để bảo vệ các tài nguyên cơ sở dữ liệu tránh khỏi các tấn công nền tảng, cần thiết phải kết hợp giữa việc cập nhật (các bản vá) phần mềm thường xuyên và sử dụng hệ thống ngăn chặn xâm nhập (IPS). Các nhà cung cấp luôn đưa ra những bản vá cho các điểm yếu trong nền tảng cơ sở dữ liệu. Tuy nhiên, các doanh nghiệp cập nhật và thực thi các phần mềm này theo định kỳ. Khi đó, giữa các chu kỳ cập nhật này, cơ sở dữ liệu không được bảo vệ. Thêm vào đó, vấn đề tương thích đôi khi không cho phép các cập nhật phần mềm này được hoàn chỉnh. Để giải quyết những vấn đề này, cần phải sử dụng IPS. Như mô tả ở trên, IPS kiểm tra lưu lượng cơ sở dữ liệu và xác định được những tấn công nhắm vào các điểm yếu đã biết.

SecureSphere bảo vệ nền tảng bằng IPS. IPS có khả năng bảo vệ cơ sở dữ liệu chống lại loại sâu và tấn công nền tảng khác. Trong thực tế, IPS của SecureSphere thậm chí có thể bảo vệ được cơ sở dữ liệu chống lại những điểm yếu mà các nhà cung cấp nền tảng cơ sở dữ liệu không công bố.

5. SQL Injection

Trong một tấn công SQL Injection, kẻ tấn công thường chèn (hay “tiêm”) các mệnh đề cơ sở dữ liệu bất hợp pháp vào một nguồn dữ liệu SQL dễ bị tổn thương. Thường nguồn dữ liệu đích bao gồm các thủ tục được lưu và các tham số đầu vào của ứng dụng Web. Các nguồn này bị tiêm vào những mệnh đề bất hợp pháp, sau đó chúng được truyền tới cơ sở dữ liệu và được xử lý tại

đây. Với SQL Injection, kẻ tấn công có thể đạt được truy nhập không giới hạn tới toàn bộ cơ sở dữ liệu.

Ngăn chặn SQL Injection

Có thể sử dụng kết hợp ba kỹ thuật để chống SQL Injection một cách hiệu quả là: IPS, kiểm soát truy nhập mức truy vấn và sự tương quan sự kiện. IPS có thể xác định các thủ tục được lưu chứa điểm yếu hoặc những chuỗi SQL injection. Tuy nhiên, một mình IPS thì không đáng tin cậy. Các nhà quản trị an toàn nếu chỉ tin cậy vào một mình IPS sẽ luôn nhận được hàng loạt các cảnh báo là SQL injection “có thể đang tồn tại”. Tuy nhiên, bằng cách tương quan một dấu hiệu của SQL injection với một xâm nhập vào kiểm soát truy nhập mức truy vấn thì có thể xác định được tấn công với độ chính xác cao.

SecureSphere tích hợp Hồ sơ động, IPS và công nghệ Phê chuẩn tấn công tương quan (Correlated Attack Validation) để xác định SQL injection với độ chính xác rất cao:

- Hồ sơ động phát ra kiểm soát truy nhập mức truy vấn bằng cách tạo cho mỗi người dùng một hồ sơ và các mẫu truy vấn thông thường của ứng dụng. Một truy vấn bất kỳ (của tấn công SQL injection) không phù hợp với hồ sơ của người dùng hay mẫu của ứng dụng đã được thiết lập trước đó, sẽ bị phát hiện ngay lập tức.

- IPS của SecureSphere chứa từ điển dấu hiệu cơ sở dữ liệu duy nhất, từ điển này được thiết kế đặc biệt để xác định các thủ tục được lưu chứa điểm yếu và các chuỗi SQL injection.

- Công nghệ Phê chuẩn tấn công tương quan sẽ làm nhiệm vụ tương quan, xâu chuỗi lại các xâm phạm an toàn được phát hiện từ các tầng của SecureSphere. Bằng khả năng xâu chuỗi nhiều xâm phạm từ cùng một người dùng, SecureSphere có thể phát hiện được SQL injection với độ chính xác rất cao mà một tầng phát hiện xâm nhập trong SecureSphere không thể làm được.

Xét một tấn công SQL injection vào thủ tục được lưu được chỉ ra ở bên dưới:

```
exec ctxsys.driload.validate_stmt ('grant dba to scott')
```

Trong tấn công này, kẻ tấn công (scott) đang cố gắng gán các đặc quyền quản trị cơ sở dữ liệu cho chính mình bằng cách nhúng một toán tử “grant” vào một thủ tục được lưu chứa điểm yếu. SecureSphere có thể kiểm soát tấn công này tùy thuộc vào hai trường hợp sau:

- Thủ tục được lưu chứa điểm yếu này không được yêu cầu cho các hoạt động của doanh nghiệp. Như vậy, trong trường hợp lý tưởng là các thủ tục được lưu chứa điểm yếu này không được sử dụng bởi bất kỳ người dùng hay ứng dụng nào của doanh nghiệp. Khi đó IPS của SecureSphere có đủ khả năng nhận dạng một cách chính xác và chặn (tùy chọn) tất cả các trường hợp của tấn công này. Những hoạt động công việc thông thường sẽ không chứa những chuỗi ký tự phức tạp như trên (...ctxsys.driload...).

- Thủ tục được lưu chứa điểm yếu là một phần trong một số hoạt động của doanh nghiệp. Trong trường hợp này, đầu tiên SecureSphere đưa ra cảnh báo cho các nhà quản trị an toàn cho việc sử dụng các thủ tục này. Sau đó, công nghệ Phê chuẩn tấn công tương quan có thể tương quan, xâu chuỗi các trường hợp sử dụng của dấu hiệu tấn công này với một loạt những người dùng và ứng dụng sử dụng nó. Nếu một người dùng bất kỳ không được phép đăng cố gắng sử dụng thủ tục này thì SecureSphere có thể phát ra cảnh báo, thậm chí là chặn lại yêu cầu của người dùng đó.

6. Lợi dụng dấu vết kiểm toán yếu (Weak Audit Trail)

Đối với bất kỳ hoạt động nào liên quan đến cơ sở dữ liệu, cần thiết phải ghi lại một cách tự động tất cả các giao dịch cơ sở dữ liệu nhạy cảm và/hoặc các giao dịch bất thường. Khi chính sách kiểm toán cơ sở dữ liệu yếu sẽ dẫn đến những rủi ro nghiêm trọng cho tổ chức với nhiều mức độ khác nhau.

Các cơ chế kiểm toán là tuyến hàng rào bảo vệ cơ sở dữ liệu cuối cùng. Nếu kẻ tấn công có thể phá vỡ các hàng rào khác thì cơ chế kiểm toán dữ liệu vẫn có thể xác định sự tồn tại của một xâm nhập sau những hành động của kẻ tấn công trước đó. Những vết kiểm toán thu được tiếp tục có thể dùng để xác định người dùng nào vừa thực hiện các hành động này, đồng thời qua vết kiểm toán có thể thực hiện phục hồi lại hệ thống.

Cơ chế kiểm toán yếu đồng nghĩa với việc hệ thống không thể ghi lại đầy đủ những hành động của người dùng, dẫn đến việc không thể phát hiện và ngăn chặn kịp thời những hành động tấn công của người dùng hoặc một nhóm người dùng. Do vậy, kiểm toán là cơ chế quan trọng mà mọi hệ thống cần có và cần thiết phải có cơ chế kiểm toán mạnh đảm bảo an toàn thông tin hay an toàn cơ sở dữ liệu cho mọi hệ thống.

- *Rủi ro về luật*: các tổ chức có cơ chế kiểm toán cơ sở dữ liệu yếu sẽ thấy mình lạc lõng với các yêu cầu về luật của chính phủ.

- *Sự ngăn chặn*: Tương tự như hệ thống máy quay ghi lại khuôn mặt của những người đi vào nhà băng, các cơ chế kiểm toán cơ sở dữ liệu cũng nhằm phát hiện kẻ tấn công nhờ việc ghi lại các vết kiểm toán cơ sở dữ liệu. Dựa trên những vết kiểm toán này, điều tra viên có thể phát hiện ra đâu là hành vi của một kẻ xâm nhập cơ sở dữ liệu.

- *Sự phát hiện và phục hồi*: Các cơ chế kiểm toán là tuyến hàng rào bảo vệ cơ sở dữ liệu cuối cùng. Nếu kẻ tấn công có thể phá vỡ các hàng rào khác thì cơ chế kiểm toán dữ liệu vẫn có thể xác định sự tồn tại của một xâm nhập sau những hành động của kẻ tấn công trước đó. Những vết kiểm toán thu được tiếp tục có thể dùng để xác định người dùng nào vừa thực hiện các hành động này, đồng thời qua vết kiểm toán có thể thực hiện phục hồi lại hệ thống.

7. Từ chối dịch vụ (*Denial of Service*)

Từ chối dịch vụ (DOS) là một loại tấn công trong đó các truy nhập của người dùng hợp pháp vào các ứng dụng mạng hay vào dữ liệu sẽ bị từ chối. Các điều kiện từ chối dịch vụ có thể được tạo ra qua nhiều kỹ thuật (nhiều trong số đó liên quan đến các điểm yếu nền tảng). Ví dụ, tấn công DOS có thể dựa trên điểm yếu nền tảng cơ sở dữ liệu để phá hủy một máy chủ. Ngoài ra còn có một số kỹ thuật DOS phổ biến khác như: sửa đổi dữ liệu, làm lụt mạng (network flooding), và làm quá tải tài nguyên máy chủ (bộ nhớ, CPU,...). Trong đó, làm quá tải tài nguyên là một kỹ thuật phổ biến trong môi trường cơ sở dữ liệu.

Ngăn chặn tấn công từ chối dịch vụ

Để ngăn chặn tấn công này, cần yêu cầu bảo vệ ở nhiều mức khác nhau, như: mức mạng, mức ứng dụng và mức cơ sở dữ liệu. Ở phần này, ta tập trung vào các bảo vệ mức cơ sở dữ liệu. Trong đó, một số kỹ thuật bảo vệ mức cơ sở dữ liệu để chống tấn công DOS bao gồm: kiểm soát tốc độ kết nối, kiểm soát giao thức, kiểm soát truy nhập truy vấn, và kiểm soát thời gian phản hồi.

Các biện pháp bảo vệ chống lại DOS của SecureSphere:

Kiểm soát kết nối: nhằm giảm thiểu việc tài nguyên máy chủ bị quá tải, bằng cách hạn chế: tốc độ kết nối, tốc độ truy vấn và một số tham biến khác cho mỗi người dùng cơ sở dữ liệu.

Phê chuẩn giao thức: nhằm ngăn chặn kẻ tấn công khai thác các điểm yếu phần mềm đã biết để tạo tấn công DOS. Ví dụ, tràn bộ đệm (buffer overflow) là một điểm yếu nền tảng phổ biến, mà từ đó kẻ tấn công có thể khai thác để phá hủy các máy chủ cơ sở dữ liệu.

Hỗ trợ động: tự động cung cấp kiểm soát truy nhập truy vấn để phát hiện các truy vấn bất hợp pháp dẫn đến tấn công DOS.

Quyết định thời gian phản hồi: các tấn công DOS vào cơ sở dữ liệu được thiết kế để làm quá tải các tài nguyên máy chủ, dẫn tới cản trở các phản hồi cơ sở dữ liệu. Đặc tính trên của SecureSphere sẽ phát hiện những sự cản trở đối với cả các phản hồi truy vấn của người dùng và các phản hồi của toàn hệ thống.

8. Lợi dụng các điểm yếu trong giao thức giao tiếp cơ sở dữ liệu (Database Communication Protocol Vulnerabilities)

Các nhà cung cấp cơ sở dữ liệu phát hiện ra một số lượng ngày càng tăng các điểm yếu an toàn trong các giao thức giao tiếp cơ sở dữ liệu. Chẳng hạn 4 trong 7 điểm yếu về giao thức trong IBM DB2 đã được cố định và sửa chữa, 11 trong số 23 điểm yếu cơ sở dữ liệu liên quan đến giao thức đã được cố định trong Oracle gần đây. Sâu SQL Slammer2 cũng là khe hở trong giao thức của Microsoft SQL Server đã gây ra tấn công từ chối dịch vụ. Ngoài ra, các hoạt động của giao thức không hề được lưu lại trong các vết kiểm toán cho nên càng gây khó khăn cho việc phát hiện lỗi và phát hiện các tấn công.

Ngăn chặn tấn công giao thức giao tiếp cơ sở dữ liệu

Có thể giải quyết tấn công này bằng công nghệ phê duyệt giao thức. Công nghệ này sẽ phân tích lưu lượng cơ sở dữ liệu cơ bản và so sánh nó với lưu lượng thông thường. Trong trường hợp lưu lượng hiện tại không phù hợp với lưu lượng cơ sở dữ liệu như mong đợi thì hệ thống sẽ phát ra cảnh báo, thậm chí là chặn luôn hành động đang thực hiện.

9. Lợi dụng sự xác thực yếu (Weak Authentication)

Trong một hệ thống chưa lược đồ xác thực yếu, kẻ tấn công dễ dàng có thể chiếm lấy định danh của những người dùng cơ sở dữ liệu hợp pháp bằng cách lấy cắp thẻ đăng nhập của họ. Kẻ tấn công có thể sử dụng nhiều chiến lược khác nhau để lấy được thẻ đăng nhập của người dùng hợp pháp, chẳng hạn:

- *Kỹ thuật Brute force*: kẻ tấn công sẽ gõ lặp đi lặp lại tổ hợp username/password cho đến khi tìm ra một cặp hợp lệ. Thường kẻ tấn công sử dụng một chương trình đoán mật khẩu tự động để làm tăng nhanh quá trình brute force này.

- *Kỹ thuật Social Engineering*: kẻ tấn công lợi dụng những khuynh hướng tự nhiên mà con người tin theo, từ đó làm cho người dùng tin tưởng và đưa ra thẻ đăng nhập của họ. Ví dụ, kẻ tấn công có thể gọi điện thoại và thể hiện rằng mình là một người quản lý công nghệ thông tin, yêu cầu người dùng cung cấp thẻ đăng nhập cho mục đích “bảo trì hệ thống”.

- *Kỹ thuật Lấy cắp thẻ đăng nhập trực tiếp*: kẻ tấn công có thể lấy cắp trực tiếp các thẻ đăng nhập của người dùng bằng cách sao chép các ghi chú ngắn của thẻ hoặc sử dụng file mật khẩu,...

Để chống tấn công này, chúng ta cần thực hiện một số cơ chế:

- *Xác thực mạnh*: Để ngăn chặn kẻ tấn công lợi dụng xác thực yếu, hệ thống cần sử dụng các chính sách và công nghệ xác thực mạnh nhất. Nên sử dụng cơ chế xác thực hai nhân tố (như: token, chứng chỉ, sinh trắc,...). Tuy nhiên, các cơ chế xác thực hai nhân tố thường có chi phí đắt và khó sử dụng hơn nên trong thực tế chúng được sử dụng không nhiều. Với trường hợp này,

cần sử dụng chính sách xác thực mạnh cho username/password (như: độ dài tối thiểu, đa dạng ký tự, khó đoán,...).

- *Tích hợp danh bạ*: Để các cơ chế xác thực mạnh được sử dụng rộng rãi và dễ dàng, nên tích hợp chúng với cơ sở danh bạ của doanh nghiệp. Cơ sở danh bạ cho phép một người dùng có thể sử dụng một tập thẻ đăng nhập cho nhiều cơ sở dữ liệu và ứng dụng. Từ đó làm cho hệ thống xác thực hai nhân tố được sử dụng hiệu quả hơn và người dùng có thể dễ dàng ghi nhớ việc cần phải thay đổi mật khẩu thường xuyên.

10. Lợi dụng sơ hở của dữ liệu dự phòng (Backup Data Exposure)

Chúng ta thấy rằng, trong công tác sao lưu, dự phòng cơ sở dữ liệu, người thực hiện thường lơ là không bảo vệ các phương tiện lưu trữ cơ sở dữ liệu dự phòng một cách đầy đủ. Kết quả là, có rất nhiều băng đĩa sao lưu cơ sở dữ liệu và các đĩa cứng của nhiều hệ thống bị đánh cắp.

Ngăn chặn sơ hở dữ liệu dự phòng

Tất cả các bản sao cơ sở dữ liệu cần thiết phải được mã hóa. Một vài nhà cung cấp khẳng định rằng, các sản phẩm DBMS tương lai của họ sẽ hỗ trợ các bản sao cơ sở dữ liệu được mã hóa.

1.4 CÁC YÊU CẦU BẢO VỆ CƠ SỞ DỮ LIỆU

Bảo vệ cơ sở dữ liệu khỏi các hiểm họa, có nghĩa là bảo vệ tài nguyên, đặc biệt là dữ liệu khỏi các thảm họa, hoặc truy nhập trái phép. Các yêu cầu bảo vệ cơ sở dữ liệu gồm:

1.4.1 Bảo vệ chống truy nhập trái phép

Đây là một vấn đề cơ bản, bao gồm trao quyền truy nhập cơ sở dữ liệu cho người dùng hợp pháp. Yêu cầu truy nhập của ứng dụng, hoặc người dùng phải được DBMS kiểm tra. Kiểm soát truy nhập cơ sở dữ liệu phức tạp hơn kiểm soát truy nhập file. Việc kiểm soát cần tiến hành trên các đối tượng dữ liệu ở mức thấp hơn mức file (chẳng hạn như các bản ghi, các thuộc tính và các giá trị). Dữ liệu trong cơ sở dữ liệu thường có quan hệ với nhau về ngữ

nghĩa, do đó cho phép người sử dụng có thể biết được giá trị của dữ liệu mà không cần truy nhập trực tiếp, bằng cách suy diễn từ các giá trị đã biết.

1.4.2 Bảo vệ chống suy diễn

Suy diễn là khả năng có được các thông tin bí mật từ những thông tin không bí mật. Đặc biệt, suy diễn ảnh hưởng tới các cơ sở dữ liệu thống kê, trong đó người dùng không được phép dò xét thông tin của các cá thể khác từ các dữ liệu thống kê đó.

1.4.3 Bảo vệ toàn vẹn cơ sở dữ liệu

Yêu cầu này bảo vệ cơ sở dữ liệu khỏi các truy nhập trái phép mà có thể dẫn đến việc thay đổi nội dung dữ liệu. Các lỗi, virus, hỏng hóc trong hệ thống có thể gây hỏng dữ liệu. DBMS đưa ra dạng bảo vệ này, thông qua các kiểm soát về sự đúng đắn của hệ thống, các thủ tục sao lưu, phục hồi và các thủ tục an toàn đặc biệt.

Để duy trì tính tương thích của cơ sở dữ liệu, mỗi giao tác phải là một đơn vị tính toán tin cậy và tương thích.

Hệ thống khôi phục (*recovery system*) sử dụng nhật ký. Với mỗi giao tác, nhật ký ghi lại các phép toán đã được thực hiện trên dữ liệu (chẳng hạn như *read, write, delete, insert*), cũng như các phép toán điều khiển giao tác (chẳng hạn như *commit, abort*), cả giá trị cũ và mới của các bản ghi kéo theo. Hệ thống phục hồi đọc file nhật ký để xác định giao tác nào bị huỷ bỏ và giao tác nào cần phải thực hiện lại. Huỷ một giao tác có nghĩa là phục hồi lại giá trị cũ của mỗi phép toán trên bản ghi kéo theo. Thực hiện lại giao tác có nghĩa là cập nhật giá trị mới của mỗi phép toán vào bản ghi kéo theo.

Các thủ tục an toàn đặc biệt bảo vệ dữ liệu không bị truy nhập trái phép. Xây dựng mô hình, thiết kế và thực hiện các thủ tục này là một trong các mục tiêu an toàn cơ sở dữ liệu.

Toàn vẹn dữ liệu thao tác

Yêu cầu này đảm bảo tính tương thích lôgíc của dữ liệu khi có nhiều giao tác thực hiện đồng thời.

Bộ quản lý tương tranh trong DBMS đảm bảo tính chất khả tuần tự và cô lập của các giao tác. Khả tuần tự có nghĩa là kết quả của việc thực hiện đồng thời một tập hợp các giao tác giống với việc thực hiện tuần tự các giao tác này. Tính cô lập để chỉ sự độc lập giữa các giao tác, tránh được hiệu ứng Domino, trong đó việc huỷ bỏ một giao tác dẫn đến việc huỷ bỏ các giao tác khác (theo kiểu thác đổ).

Vấn đề đảm bảo truy nhập đồng thời vào cùng một thực thể dữ liệu, từ các giao tác khác nhau, nhưng không làm ảnh hưởng đến tính tương thích của dữ liệu, được giải quyết bằng các kỹ thuật khoá.

Các kỹ thuật khoá và giải phóng khoá được thực hiện theo nguyên tắc: khoá các mục dữ liệu trong một khoảng thời gian cần thiết để thực hiện phép toán và giải phóng khoá khi phép toán đã hoàn tất. Tuy nhiên kỹ thuật này không đảm bảo tính khả tuần tự. Nhược điểm này được khắc phục bằng cách sử dụng kỹ thuật khoá hai pha.

Toàn vẹn ngữ nghĩa của dữ liệu

Yêu cầu này đảm bảo tính tương thích lôgíc của các dữ liệu bị thay đổi, bằng cách kiểm tra các giá trị dữ liệu có nằm trong khoảng cho phép hay không. Các hạn chế (trên các giá trị dữ liệu) được biểu diễn như là các ràng buộc toàn vẹn. Các ràng buộc có thể được xác định trên toàn bộ cơ sở dữ liệu hoặc là cho một số các giao tác.

1.4.4 Khả năng lưu vết và kiểm tra

Yêu cầu này bao gồm khả năng ghi lại mọi truy nhập tới dữ liệu (với các phép toán *read* và *write*). Khả năng kiểm tra và lưu vết đảm bảo tính toàn vẹn dữ liệu vật lý và trợ giúp cho việc phân tích dãy truy nhập vào cơ sở dữ liệu.

1.4.5 Xác thực người dùng

Yêu cầu này thực sự cần thiết để xác định tính duy nhất của người dùng. Định danh người dùng làm cơ sở cho việc trao quyền. Người dùng được phép truy nhập dữ liệu, khi hệ thống xác định được người dùng này là hợp pháp.

1.4.6 Quản lý và bảo vệ dữ liệu nhạy cảm

Có những cơ sở dữ liệu chứa nhiều dữ liệu nhạy cảm (là những dữ liệu không nên đưa ra công bố công khai). Có những cơ sở dữ liệu chỉ chứa các dữ liệu nhạy cảm, chẳng hạn như dữ liệu quân sự, còn có các cơ sở dữ liệu mang tính công cộng, chẳng hạn như các cơ sở dữ liệu của thư viện.

Các cơ sở dữ liệu bao gồm cả thông tin nhạy cảm và thông tin thường cần phải có các chính sách quản lý phức tạp hơn. Một mục dữ liệu là nhạy cảm khi chúng được người quản trị cơ sở dữ liệu (DBA) khai báo là nhạy cảm.

Kiểm soát truy nhập vào các cơ sở dữ liệu bao hàm: bảo vệ tính tin cậy của dữ liệu nhạy cảm và chỉ cho phép người dùng hợp pháp truy nhập vào. Những người dùng này được trao một số quyền thao tác nào đó trên dữ liệu và không được phép lan truyền chúng. Do vậy, người dùng có thể truy nhập vào các tập con dữ liệu nhạy cảm.

1.4.7 Bảo vệ nhiều mức

Bảo vệ nhiều mức bao gồm một tập hợp các yêu cầu bảo vệ. Thông tin có thể được phân loại thành nhiều mức khác nhau, ví dụ các cơ sở dữ liệu quân sự cần được phân loại chi tiết hơn (mịn hơn) các cơ sở dữ liệu thông thường, có thể có nhiều mức nhạy cảm khác nhau. Mục đích của bảo vệ nhiều mức là phân loại các mục thông tin khác nhau, đồng thời phân quyền cho các mức truy nhập khác nhau vào các mục riêng biệt. Một yêu cầu nữa đối với bảo vệ nhiều mức là khả năng gán mức cho các thông tin.

1.4.8 Sự hạn chế

Mục đích của việc hạn chế là tránh chuyển các thông tin không mong muốn giữa các chương trình trong hệ thống, ví dụ chuyển dữ liệu quan trọng tới các chương trình không có thẩm quyền. Các kênh được phép cung cấp thông tin thông qua các hoạt động được phép, như soạn thảo hay biên dịch một file. Kênh bộ nhớ là các vùng bộ nhớ, nơi một chương trình có thể lưu giữ dữ liệu, các chương trình khác cũng có thể đọc dữ liệu này. Kênh ngầm là kênh truyền thông dựa trên việc sử dụng tài nguyên mà không có ý định truyền thông giữa các tiến trình của hệ thống.

1.5 CÂU HỎI ÔN TẬP

1. Cơ sở dữ liệu là gì? Nêu tầm quan trọng của cơ sở dữ liệu?
2. Hệ quản trị cơ sở dữ liệu là gì? Nêu một số hệ quản trị thông dụng.
3. Phân biệt giữa siêu dữ liệu (metadata) và thể hiện dữ liệu (data instance)
4. Nêu các bước thiết kế một cơ sở dữ liệu
5. Nêu và viết một số câu lệnh SQL.
6. Nêu mười hiểm họa hàng đầu đối với cơ sở dữ liệu.
7. Nêu các yêu cầu đối với việc bảo vệ cơ sở dữ liệu

CHƯƠNG 2. CÁC MÔ HÌNH VÀ CHÍNH SÁCH AN TOÀN

Trong chương này, chúng ta sẽ tìm hiểu về các mô hình và chính sách an toàn cơ sở dữ liệu, điển hình như: mô hình an toàn tùy ý và mô hình an toàn bắt buộc. Trong mỗi mô hình đều có phần chỉ ra những hạn chế mà các mô hình chưa đáp ứng được. Phần cuối chương giới thiệu một số mô hình an toàn khác.

2.1 MỘT SỐ KHÁI NIỆM CƠ BẢN

Trước hết, chúng ta tìm hiểu một số khái niệm cơ bản được sử dụng trong chương này.

2.1.1 Mô hình an toàn

Mô hình an toàn là một mô hình khái niệm mức cao, độc lập phần mềm và xuất phát từ các đặc tả yêu cầu của tổ chức để mô tả nhu cầu bảo vệ của một hệ thống.

Trong mỗi mô hình an toàn, có hai khái niệm cần quan tâm, đó là: chủ thể (subject) và đối tượng (object). Chủ thể là một thực thể chủ động, chẳng hạn: người dùng và các tiến trình. Đối tượng là các thực thể bị động, như: các file, bảng, khung nhìn, segment, printer,... Trong cơ sở dữ liệu đa mức, chúng ta còn quan tâm đến các đối tượng mịn hơn như: các hàng, các cột, các ô.

2.1.2 Chính sách an toàn

Chính sách an toàn của hệ thống là các quy tắc, các hướng dẫn ở mức cao, có liên quan đến việc thiết kế và quản lý hệ thống trao quyền. Chính sách an toàn là những phát biểu mức tổng quát về an toàn thông tin từ phía nhà quản lý định nghĩa các quy tắc, trong đó quy định những truy nhập nào được phép và những truy nhập nào bị từ chối.

Mục đích của chính sách an toàn là xây dựng ra mục tiêu an toàn chung cho ba vấn đề bí mật, toàn vẹn, và sẵn sàng trong một hệ thống cụ thể. Đây là 3 vấn đề phát sinh trong tất cả các hệ thống thông tin thực tế, ví dụ như một

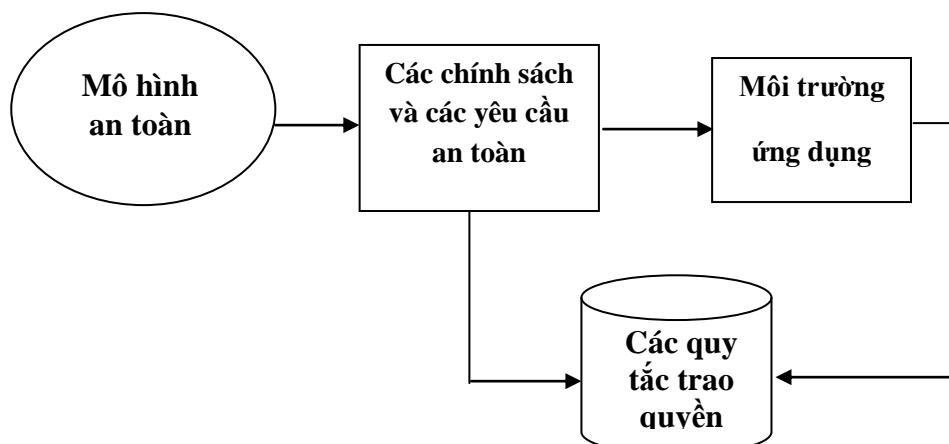
hệ thống bí mật ngăn chặn việc nhân viên có thể tìm ra lương của giám đốc. Toàn vẹn liên quan đến việc ngăn ngừa một nhân viên tự thay đổi mức lương của mình trong CSDL, sẵn sàng liên quan đến việc tiền lương được đảm bảo trả đúng hạn, đúng giá trị theo quy định của pháp luật. Tương tự, trong Quân sự, chẳng hạn tính bí mật liên quan đến việc ngăn chặn việc lộ thông tin các mục tiêu, tọa độ của một tên lửa, toàn vẹn quan tâm đến việc quân địch thay đổi tọa độ mục tiêu hay không, và sẵn sàng liên quan đến việc đảm bảo tên lửa chỉ phóng khi có lệnh.

Bất kỳ hệ thống nào cũng có ba vấn đề này. Có sự khác biệt về tầm quan trọng của những vấn đề trong một hệ thống cụ thể. Cả hai ngành thương mại và quân sự đều có nhu cầu cao về toàn vẹn. Nhưng riêng về quân sự thì tính bí mật và sẵn sàng thường chặt chẽ hơn trong thương mại.

Trong hệ quản trị Oracle, ta có thể thu được một danh sách tất cả các chính sách bằng cách truy vấn vào khung nhìn DBA_POLICIES. Ngoài ra, các khung nhìn ALL_POLICIES và USER_POLICIES cung cấp thông tin về các chính sách đã được định nghĩa.

2.1.3 Quy tắc trao quyền

Quá trình thiết kế một hệ thống an toàn phải đưa ra được một mô hình an toàn và mô hình này hỗ trợ người trao quyền khi ánh xạ các yêu cầu an toàn, chính sách an toàn vào các quy tắc trao quyền, như hình vẽ dưới đây:



Hình 2.1 Thiết kế các quy tắc trao quyền

Nhiệm vụ của người cấp quyền là chuyển đổi các yêu cầu và các chính sách an toàn mà tổ chức đưa ra, thành các quy tắc trao quyền. Từ mô hình an toàn mức cao (phần thiết kế có cấu trúc), người trao quyền xây dựng nên các chính sách an toàn của tổ chức nhằm mô tả các yêu cầu của tổ chức. Đồng thời kết hợp với các yêu cầu của ứng dụng thực tế mà, người trao quyền xây dựng nên các quy tắc trao quyền thực trong hệ thống.

Các quy tắc trao quyền được biểu diễn đúng với môi trường phần mềm/phần cứng của hệ thống bảo vệ và phải phù hợp với các chính sách an toàn đã đưa ra. Ví dụ, trong hệ quản trị SQL Server, Oracle.

Một ví dụ về ma trận quyền được trình bày trong bảng 2.1, trong đó $R=Read$, $W=Write$, $EXC=Execute$, $CR=Create$, và $DEL>Delete$. Đây là một trường hợp kiểm soát truy nhập đơn giản dựa vào tên đối tượng, được gọi là truy nhập phụ thuộc tên (*name-dependent access*). Các quyền có thể bao gồm nhiều quy tắc an toàn phức tạp hơn, chúng xác định các ràng buộc truy nhập giữa chủ thể và đối tượng.

Một tân từ (*predicate*) cũng có thể được xem là biểu thức của một số biến hệ thống, chẳng hạn như ngày, giờ và nguồn truy vấn. Tân từ là cơ sở cho kiểm soát phụ thuộc ngữ cảnh (*context-dependent control*). Một ví dụ về kiểm soát phụ thuộc ngữ cảnh là không thể truy nhập vào thông tin được phân loại thông qua một đăng nhập từ xa (*remote login*), hoặc chỉ cập nhật thông tin về lương vào thời điểm cuối của năm. Kiểm soát phụ thuộc nội dung (*content-dependent control*) nằm ngoài phạm vi của các hệ điều hành, nó do DBMS cung cấp. Với kiểm soát phụ thuộc ngữ cảnh (*context-dependent control*), một phần do hệ điều hành cung cấp, một phần do DBMS cung cấp.

	Đối tượng		
Chủ thẻ	File F1	File F2	File F3
Người dùng 1	R,W	EXEC	EXEC
Người dùng 2	-	-	CR, DEL
Chương trình P1	R,W	R	-

Bảng 2.1 Ma trận quyền

Các quy tắc an toàn cũng nên xác định các kết hợp dữ liệu không được phép, cần phải xem độ nhạy cảm của dữ liệu có tăng lên sau khi kết hợp hay không. Ví dụ, người dùng có thể được phép đọc tên và các giá trị lương của nhân viên một cách riêng lẻ, nhưng không được phép đọc kết hợp "tên - lương", nếu không họ có thể liên hệ lương với từng nhân viên cụ thể.

2.2 CÁC MÔ HÌNH AN TOÀN CƠ SỞ DỮ LIỆU

2.2.1 Kiểm soát truy nhập trong các hệ thống hiện tại

Mục đích của kiểm soát truy nhập là để đảm bảo rằng người dùng chỉ được phép thực hiện các hoạt động trên cơ sở dữ liệu mà người dùng được cấp quyền. Kiểm soát truy nhập trong các hệ thống thông tin là đảm bảo mọi truy nhập trực tiếp vào các đối tượng của hệ thống tuân theo các kiểu và các quy tắc đã được xác định trong chính sách bảo vệ. Kiểm soát truy nhập dựa trên việc xác định một cách chính xác một người dùng nào đó bằng thủ tục xác thực (chẳng hạn như: username, mật khẩu, mã xác thực). Việc xác thực người dùng có thể được thực hiện bởi hệ điều hành, hệ quản trị cơ sở dữ liệu hay một máy chủ xác thực riêng.

Trong phần tiếp theo, chúng ta tập trung tìm hiểu về các kiểm soát truy nhập được cung cấp trong các hệ quản trị cơ sở dữ liệu quan hệ thương mại hiện nay. Trong đó, hai loại điều khiển truy nhập phổ biến nhất là: kiểm soát truy nhập tùy ý (discretionary access control – DAC) và kiểm soát truy nhập

bắt buộc (mandatory access control - MAC). MAC là kiểm soát truy nhập được dùng cho hệ thống an toàn đa mức. Một cách gọi khác của các kiểm soát này là chính sách kiểm soát truy nhập tùy ý và chính sách kiểm soát truy nhập bắt buộc. Hai loại chính sách kiểm soát truy nhập này thuộc vào mô hình an toàn tùy ý và mô hình an toàn bắt buộc.

Các chính sách truy nhập của hệ thống nhiều mức có thể là bắt buộc hoặc tùy ý: *Kiểm soát truy nhập bắt buộc (MAC – Mandatory Access Controls)* hạn chế truy nhập của các chủ thể vào các đối tượng bằng cách sử dụng các nhãn an toàn. *Kiểm soát truy nhập tùy ý (DAC – Discretionary Access Controls)* cho phép lan truyền các quyền truy nhập từ chủ thể này đến chủ thể khác.

Các chính sách bắt buộc và tuỳ ý là không loại trừ lẫn nhau. Chúng có thể được kết hợp với nhau để tạo ra một hệ thống an toàn đầy đủ, hoàn thiện.

2.2.2 Các mô hình an toàn tùy ý

Mô hình an ninh tùy ý đã được nghiên cứu trong một thời gian dài và là nền tảng cho các hệ điều hành và các DBMS. Từ năm 1970 đến năm 1975, mô hình này đã nhận được một lượng quan tâm lớn về mặt lý thuyết. Sau thời gian này, các nghiên cứu về an toàn cơ sở dữ liệu quan hệ đã chuyển sang các kỹ thuật bảo mật khác. Tuy nhiên, sự xuất hiện của các mô hình dữ liệu tiên tiến hơn đã làm tăng thêm nhu cầu và sự quan tâm đến chính sách an toàn tùy ý.

Ví dụ về một số mô hình an toàn tùy ý như: Mô hình ma trận truy nhập (Lampson, 1971; Graham-Denning, 1973; Harrison, 1976), mô hình Take-Grant (Jones, 1976), mô hình Action-Entity (Bussolati, 1983; Fugini-Martella, 1984), mô hình của Wood-1979 như kiến trúc ANSI/SPARC đề cập đến vấn đề cấp quyền trong các cơ sở dữ liệu quan hệ lược đồ - nhiều mức,...

2.2.2.1 Kiểm soát truy nhập tùy ý

Kiểm soát truy nhập tùy ý (DAC) được xây dựng dựa trên các khái niệm của một tập các đối tượng an toàn O , một tập các chủ thể an toàn S , một tập các đặc quyền truy nhập T xác định loại truy nhập của một chủ thể đến một đối tượng nhất định và một tập các tân từ P – biểu diễn cho các quy tắc truy

nhập dựa trên ngữ cảnh. Áp dụng cho cơ sở dữ liệu quan hệ, O là một tập hữu hạn các giá trị $\{o_1, o_2, \dots, o_n\}$ đại diện cho các lược đồ quan hệ, S là một tập hữu hạn các chủ thể tiềm năng $\{s_1, \dots, s_m\}$ đại diện cho người dùng, các nhóm người dùng hoặc các phiên giao dịch đại diện cho người dùng.

Kiểu truy nhập (Privileges) là tập hợp các thao tác trên cơ sở dữ liệu như: select, insert, delete, update, execute, grant, hay revoke, và tân tử $p \in P$, xác định cửa sổ truy nhập của chủ thể $s \in S$ trên đối tượng $o \in O$. Một bộ $\langle o, s, t, p \rangle$ được gọi là *một quy tắc truy nhập (access rule)* và một hàm f được định nghĩa để xác định xem một ủy quyền $f(o, s, t, p)$ có hợp lệ hay không.

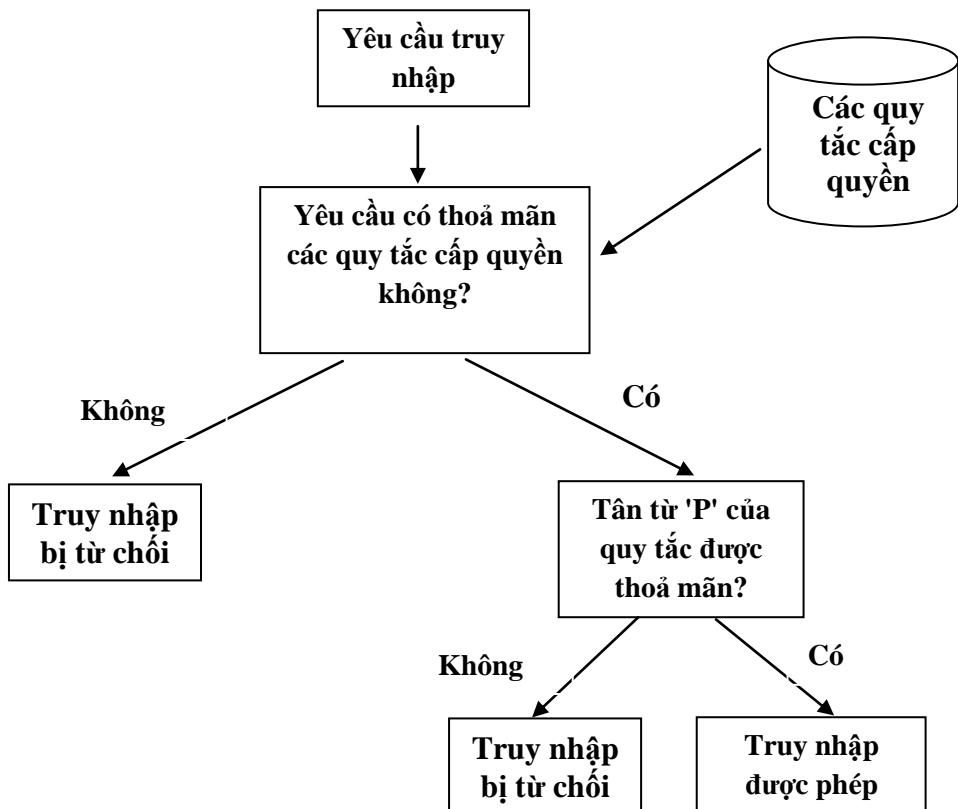
$$f: O \times S \times T \times P \rightarrow \{\text{True}, \text{False}\}.$$

Với một quy tắc truy nhập $\langle o, s, t, p \rangle$ bất kỳ, nếu $f(o, s, t, p)$ có giá trị là True thì chủ thể s có quyền truy nhập t vào đối tượng o trong phạm vi được định nghĩa bởi tân tử p .

Một tính chất quan trọng của mô hình an ninh tùy ý là sự hỗ trợ của nguyên tắc *phân quyền* hay *gán quyền*, trong đó, một quyền là phần (o, t, p) là một phần của quy tắc truy nhập. Khi một chủ thể s_i có quyền (o, t, p) thì có thể được phép gán quyền đó cho chủ thể s_j khác ($i \neq j$).

Hầu hết các hệ thống hỗ trợ DAC đều lưu trữ các quy tắc truy nhập trong một *ma trận kiểm soát truy nhập (access control matrix)*. Dạng đơn giản nhất của ma trận kiểm soát truy nhập là các hàng đại diện cho các chủ thể, các cột đại diện cho các đối tượng, giao nhau của một hàng và một cột chứa các kiểu truy nhập mà chủ thể có thể thực hiện trên đối tượng. Ví dụ về ma trận này được biểu diễn ở bảng 2.1. Ma trận kiểm soát truy nhập là cơ sở của mô hình kiểm soát truy nhập tùy ý được xây dựng bởi Lampson(1971) và sau đó được cải tiến bởi Graham và Denning(1972), sửa đổi bởi Harrison và các cộng sự (1976). Chúng ta có thể tìm hiểu chi tiết hơn mô hình này trong cuốn sách của Fernandez 1981.

Chính sách tùy ý chỉ rõ những đặc quyền mà mỗi chủ thể có thể có được trên các đối tượng và trên hệ thống (object privilege, system privilege). Các yêu cầu truy nhập được kiểm tra, thông qua một cơ chế kiểm soát tùy ý, truy nhập chỉ được trao cho các chủ thể thoả mãn các quy tắc cấp quyền hiện có.



Hình 2.2 Kiểm soát truy nhập tùy ý

Chính sách tuỳ ý dựa vào định danh của người dùng có yêu cầu truy nhập. ‘Tùy ý’ có nghĩa rằng người sử dụng có khả năng cấp phát hoặc thu hồi quyền truy nhập trên một số đối tượng. Điều này ngầm định rằng, việc phân quyền kiểm soát dựa vào quyền sở hữu (*kiểu chính sách cấp quyền dựa vào quyền sở hữu*) – như trong hệ quản trị Oracle. Tuy nhiên, quyền vẫn được người quản trị hệ thống quản lý.

Trao quyền: chính sách tuỳ ý cần các cơ chế trao quyền phức tạp hơn, nhằm tránh mất quyền kiểm soát khi lan truyền quyền từ người trao quyền, hoặc những người có trách nhiệm khác (ví dụ: trong Oracle có GRANT OPTION, ADMIN OPTION).

Thu hồi quyền: việc thu hồi quyền đã được lan truyền là một vấn đề khác. Người dùng muốn thu hồi quyền (người đã được trao quyền đó) phải có đặc quyền để thu hồi quyền. (Trong Oracle, nếu một user có GRANT OPTION, anh ta có thể thu hồi quyền đã truyền cho người khác).

Đây là một kỹ thuật phổ biến, chỉ có một vài vấn đề nghiên cứu mở. Hầu hết các DBMS thương mại đều hỗ trợ nó như: SQL Server, Oracle,...

Mô hình an toàn tùy ý được thực thi trong hầu hết các sản phẩm DBMS thương mại. Thay bởi việc gán quyền cho một người dùng lên một bảng quan hệ, thì ma trận kiểm soát truy nhập được dùng để hạn chế các hành động của người dùng trên một tập các dữ liệu có sẵn. Lưu ý rằng, cơ chế *kiểm soát truy nhập phụ thuộc dữ liệu* là một cơ chế thuộc kiểm soát truy nhập tùy ý.

2.2.2.1.1 Các chế độ kiểm soát truy nhập

Kiểm soát truy nhập có thể được áp dụng ở các mức độ chi tiết khác nhau trong hệ thống, bao gồm:

- Toàn bộ cơ sở dữ liệu
- Một tập các quan hệ (bảng CSDL)
- Một quan hệ (một bảng)
- Một vài cột của một quan hệ
- Một vài hàng của một quan hệ
- Một vài cột của một vài hàng trong một quan hệ

Kiểm soát truy nhập cũng được phân biệt trong từng hoạt động cụ thể. Điều này rất quan trọng, ví dụ như, một nhân viên được phép xem lương của mình nhưng không thể ghi hay chỉnh sửa giá trị lương đó. Trong CSDL quan hệ chế độ kiểm soát truy nhập được thể hiện bằng các câu lệnh như: SELECT, UPDATE, INSERT và DELETE:

- INSERT và DELETE được xác định trên một bảng quan hệ.
- SELECT cũng được xác định trên một bảng quan hệ. Độ chi tiết cấp quyền cho lệnh SELECT được cung cấp bởi các khung nhìn.
- UPDATE có thể được hạn chế trong một vài cột nhất định của một bảng quan hệ.

Ngoài các chế độ kiểm soát truy nhập áp dụng cho các quan hệ hoặc một phần của quan hệ, cũng có những đặc quyền mà được áp dụng đối với tài

khoản người dùng đặc biệt. Một ví dụ phổ biến là đặc quyền DBA dành cho các quản trị viên CSDL.

2.2.2.1.2 Mô hình System-R

Các mô hình cấp quyền theo cơ chế DAC hiện tại đều dựa trên mô hình System R. System R do Griffiths và Wade phát triển vào 1976, là một trong những mô hình ra đời đầu tiên cho hệ quản trị cơ sở dữ liệu quan hệ. System R dựa trên nguyên lý cấp quyền quản trị cho người sở hữu.

Các đối tượng được quản lý trong mô hình là bảng (table) và khung nhìn (view). Các chế độ truy nhập là: select, insert, update, delete, drop, index (chỉ cho table), alter (chỉ cho table). Hỗ trợ làm việc trên nhóm (group), không hỗ trợ vai trò (role). System R dùng câu lệnh GRANT để cấp quyền, có tùy chọn GRANT OPTION. Sự cấp quyền được thể hiện thông qua GRANT OPTION, nếu cấp quyền cho một người dùng bằng câu lệnh có GRANT OPTION thì người dùng đó ngoài việc có thể thực hiện quyền được cấp, còn có thể cấp quyền đó cho các người dùng khác. Một người dùng chỉ có thể cấp quyền trên một quan hệ cho trước nếu người dùng đó là người sở hữu quan hệ hoặc người dùng đó được người khác cấp quyền với câu lệnh có tùy chọn là GRANT OPTION.

Câu lệnh GRANT có cú pháp như sau:

```
GRANT Privilege List| ALL[PRIVILEGES]
ON Relation | View
TO UserList | PUBLIC
[WITH GRANT OPTION]
```

Với từng người dùng, hệ thống ghi nhận lại:

- A: tập hợp các quyền mà người dùng này sở hữu.
- B: tập hợp các quyền mà người dùng này có thể ủy quyền cho người dùng khác.

Khi một người dùng thi hành câu lệnh GRANT, hệ thống sẽ

- Tìm phần giao của tập B với tập quyền trong câu lệnh.
- Nếu phần giao là rỗng, thì câu lệnh không được thi hành.

Việc thu hồi quyền truy nhập được thực hiện qua câu lệnh REVOKE.

Câu lệnh REVOKE có cú pháp như sau:

REVOKE Privilege List| ALL[PRIVILEGES]

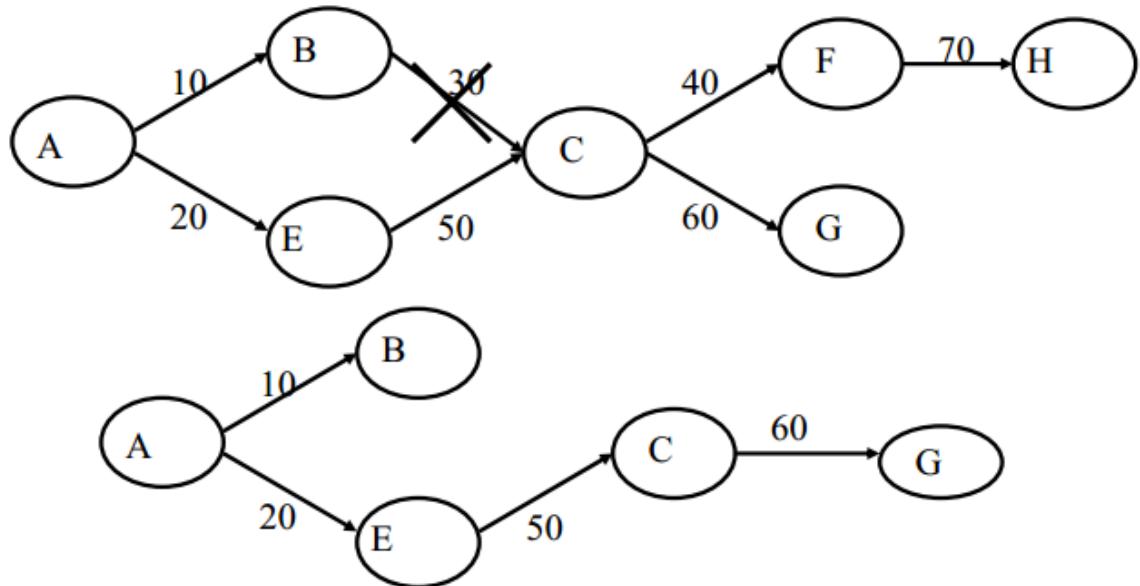
ON Relation | View

FROM UserList| PUBLIC

Ví dụ về các câu lệnh GRANT và REVOKE, bạn đọc có thể xem tiếp ở phần 2.2.1.4.3.

Câu lệnh này dùng để thu hồi các quyền đã cấp. Người dùng chỉ có thể thu hồi quyền mà người dùng đó đã cấp. Lưu ý rằng, một người dùng chỉ có thể bị thu hồi quyền truy xuất p khi tất cả những người cấp quyền p cho họ đều thu hồi quyền p lại.

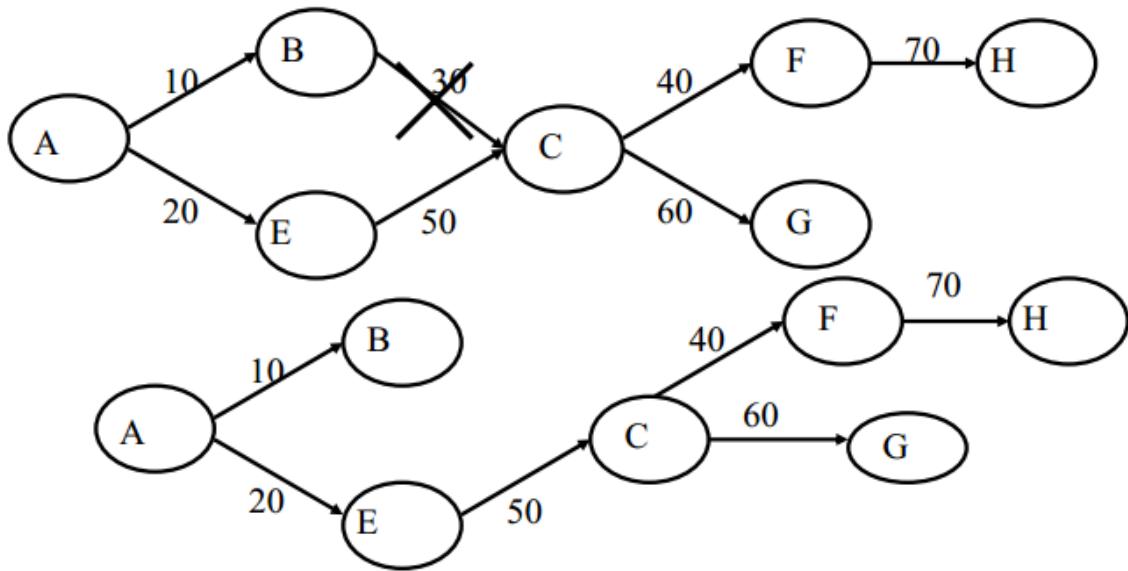
Thu hồi quyền đệ quy (recursive revocation): Khi người dùng A thu hồi quyền truy nhập của một người B thì tất cả các quyền mà B đã gán cho người khác đều bị thu hồi.



Hình 2.3 Thu hồi quyền đệ quy

Thu hồi quyền đệ quy trong System R dựa vào nhãn thời gian mỗi lần cấp quyền truy nhập cho người dùng.

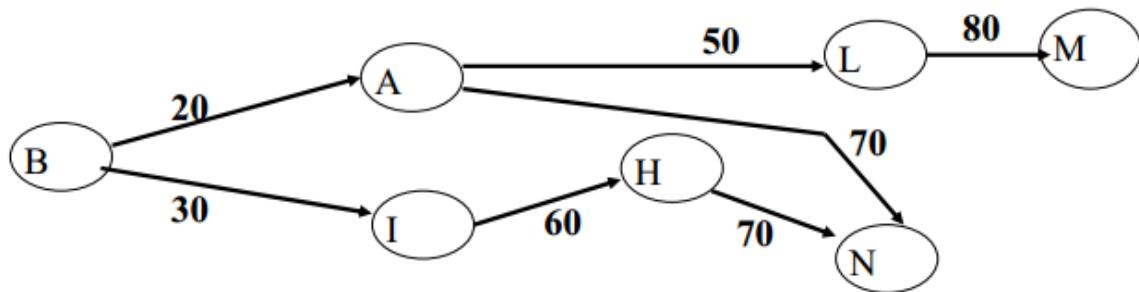
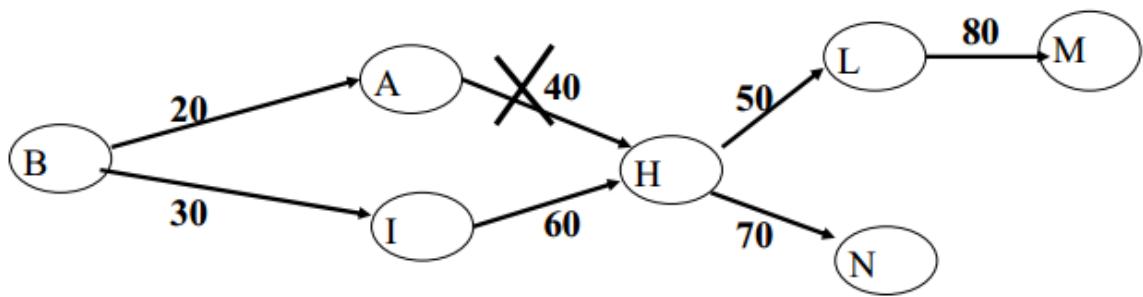
Thực tế khi một người dùng A thay đổi công việc hay vị trí thì đôi khi tổ chức chỉ muốn lấy lại quyền truy nhập của A mà không muốn lấy lại các quyền truy nhập mà A đã cấp. Một biến thể của thu hồi quyền đệ quy là không dựa vào nhãn thời gian, mục đích là để tránh thu hồi quyền đệ quy. Khi đó, nếu C bị B thu hồi quyền và C lại có quyền tương đương do người khác cấp (sau đó) thì quyền mà C cấp cho người khác vẫn được giữ.



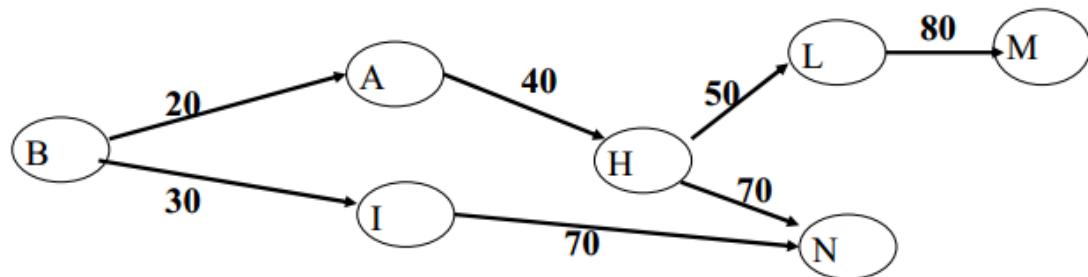
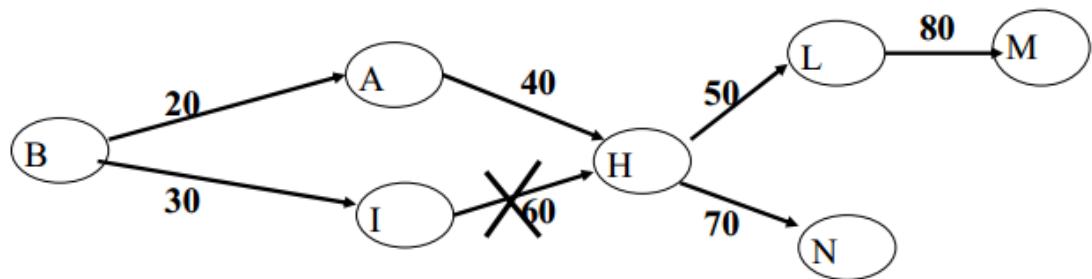
Hình 2.4 Một biến thể của thu hồi quyền đệ quy

Thu hồi quyền không đệ quy: Khi A thu hồi quyền truy nhập trên B thì tất cả quyền truy nhập mà B đã cấp cho chủ thẻ khác được thay bằng A đã cấp cho những chủ thẻ này.

Lưu ý rằng: Bởi vì B được cấp quyền truy nhập trên đối tượng từ nhiều chủ thẻ (khác A), nên không phải tất cả các quyền mà B cấp đều thay bằng A cấp. Và A được xem là người cấp thay cho B khi B sử dụng quyền A đã cấp cho B để cấp cho những chủ thẻ khác. A sẽ là người cấp các quyền mà B đã cấp sau khi B nhận quyền đó từ A có chỉ định WITH GRANT OPTION. Với những quyền B đã được cấp bởi $C \neq A$, đến lượt B cấp cho người khác thì B vẫn là người cấp các quyền này.



Hình 2.5 Thu hồi quyền không đê quy (ví dụ 1)



Hình 2.6 Thu hồi quyền không đê quy (ví dụ 2)

Lưu ý rằng với quyền mà H cấp cho L, sau khi thu hồi quyền, không được thay I như là người cấp vì quyền này được cấp trước khi I cấp quyền cho H.

2.2.2.1.3 Kiểm soát truy nhập dựa vào vai trò (Role Based Access Control – RBAC)

Hầu hết các DBMS đều hỗ trợ RBAC. RBAC có thể dùng kết hợp với DAC hoặc MAC hoặc được dùng độc lập. Đa số các DBMS chỉ hỗ trợ RBAC đơn giản.

RBAC được áp dụng vào đầu những năm 1970. Khái niệm chính của RBAC là những quyền hạn được liên kết với những vai trò (role). Khi số lượng chủ thể và đối tượng lớn thì số lượng quyền hạn có thể trở nên vô cùng lớn. Và nếu người dùng có nhu cầu cao, số lượng cấp và thu hồi quyền diễn ra thường xuyên. Do đó, nếu chúng ta cấp hoặc thu hồi từng đặc quyền cho từng người dùng thì rất tốn công sức và thời gian.

Mục đích chính của RBAC là giúp cho việc quản trị an toàn một cách dễ dàng hơn. Nhiều hệ thống điều khiển truy nhập thành công trên lĩnh vực thương mại cho các máy tính lớn trong việc quản trị an toàn, đều dựa trên vai trò. Ví dụ, một người có vai trò điều hành có thể truy nhập đến tất cả các tài nguyên nhưng không thể thay đổi được quyền truy nhập của mình và người khác; một người có vai trò nhân viên an toàn có thể truy nhập các vết kiểm toán thống kê. Việc sử dụng vai trò cũng được tìm thấy trong các hệ điều hành mạng hiện đại như Novell's NetWare hay Microsoft Windows NT.

RBAC có thể giới hạn trước các mối quan hệ vai trò – quyền hạn, làm cho việc gán người dùng đến các vai trò được xác định trước một cách dễ dàng hơn. Không có RBAC sẽ khó khăn cho việc xác định quyền hạn nào được quy định đến người dùng nào. Với RBAC, những người dùng được chỉ định những vai trò thích hợp. Điều này làm đơn giản cho việc quản lý quyền hạn trong các DBMS.

Trong một tổ chức, những chức năng công việc khác nhau được phân thành những vai trò và người dùng được chỉ định vai trò phù hợp dựa vào trách nhiệm và năng lực của họ.

Vai trò (Role) là một tập các quyền. Trong RBAC, không thực hiện cấp quyền cho từng chủ thể mà gán cho chủ thể một vai trò, khi đó chủ thể sẽ có tất cả các quyền thuộc vai trò đó.

Vai trò (Role) và nhóm (Group)

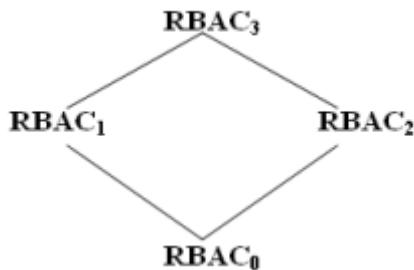
Một câu hỏi luôn được đặt ra là: Sự khác nhau giữa vai trò (Role) và nhóm (Group) là gì? Nhóm người dùng như là một đơn vị kiểm soát truy nhập thường được cung cấp bởi nhiều hệ thống kiểm soát truy nhập. Sự khác nhau chủ yếu giữa nhóm và vai trò là nhóm được đề cập như một tập hợp các người dùng mà không phải là một tập hợp các quyền. Vai trò vừa là tập hợp người dùng vừa là tập hợp các quyền. Máy chủ vai trò đóng vai trò làm cầu nối để đưa hai tập hợp này vào với nhau.

Trong nội bộ một tổ chức, các vai trò được tạo ra nhằm đảm nhận các chức năng công việc khác nhau. Mỗi vai trò được gắn liền với một số quyền hạn cho phép thao tác một số hoạt động cụ thể. Các thành viên trong lực lượng cán bộ công nhân viên (hoặc những người dùng trong hệ thống) được gán một vai trò riêng, và thông qua việc phân phối vai trò này, họ sẽ có được một số những quyền hạn cho phép họ thực hiện những chức năng cụ thể trong hệ thống.

Vì người dùng không được cấp quyền một cách trực tiếp, mà thông qua các vai trò họ được gán, nên việc quản lý quyền hạn của người dùng trở thành một việc đơn giản, và người quản trị chỉ cần gán những vai trò thích hợp cho người dùng mà thôi. Việc gán vai trò này đơn giản hóa những công việc thông thường như việc thêm một người dùng vào trong hệ thống, hay đổi phòng làm việc, chức vụ của một người dùng, khi đó ta chỉ cần thu hồi các vai trò này và gán cho họ các vai trò khác phù hợp với vị trí của họ.

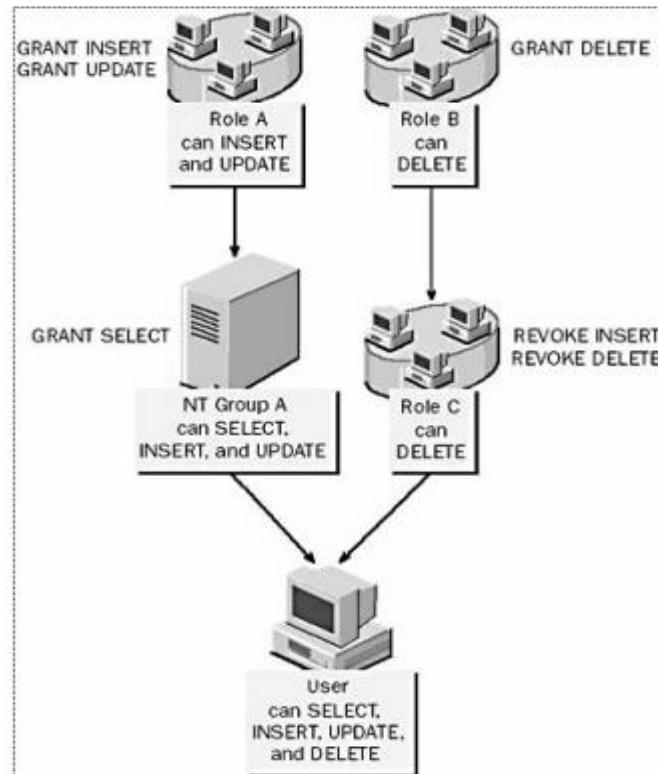
Các mô hình trong RBAC

Mô hình RBAC gồm bốn mô hình con là: RBAC0 , RBAC1 , RBAC2 , RBAC3. Trong đó, RBAC0 là nền tảng, RBAC1 thêm vào khái niệm kế thừa của hệ thống phân cấp vai trò. Một vai trò có thể kế thừa quyền hạn từ vai trò khác. RBAC2 thêm vào các ràng buộc. RBAC3 là tổng hợp của 3 mô hình trên.

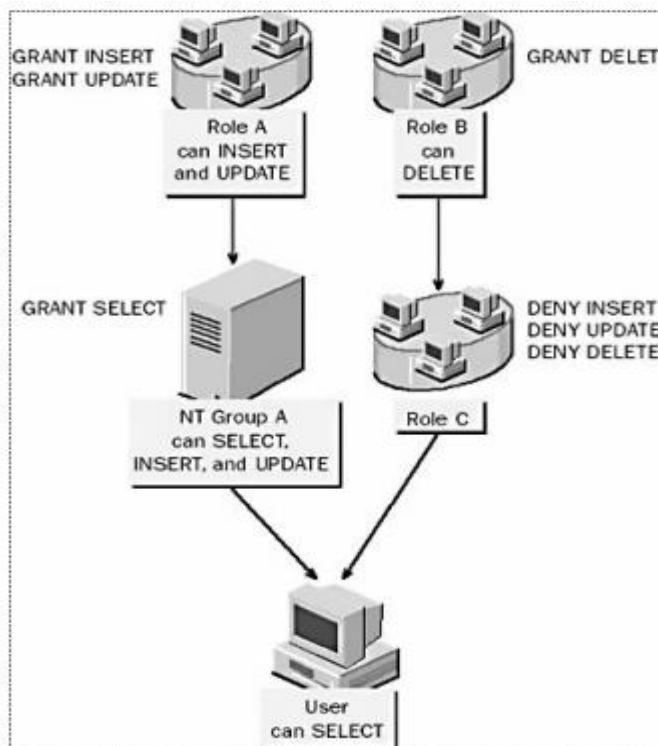


Hình 2.7 Mối quan hệ giữa các mô hình RBAC

Mô hình nền tảng RBAC0 ở dưới cùng, nó là yêu cầu tối thiểu cho bất kỳ hệ thống nào có hỗ trợ RBAC. Mô hình RBAC1, RBAC2 được phát triển từ mô hình RBAC0 nhưng có thêm các điểm đặc trưng cho từng mô hình. RBAC1 thêm vào khái niệmcủa hệ thống phân cấp vai trò (các trạng thái trong đó vai trò có thể thừa kế quyền hạn từ vai trò khác). RBAC2 thêm vào các ràng buộc (áp dụng ràng buộc để có thể thừa nhận cấu hình của các thành phần khác nhau của RBAC). RBAC1 , RBAC2 không liên quan nhau. RBAC3 là mô hình tổng hợp của ba mô hình RBAC0, RBAC1 và RBAC2.



Hình 2.8 RBAC (ví dụ 1)



Hình 2.9 RBAC (ví dụ 2)

2.2.2.1.4 Kiểm soát truy nhập phụ thuộc dữ liệu

Kiểm soát truy nhập cơ sở dữ liệu thường *phụ thuộc vào dữ liệu*. Ví dụ, một số người dùng có thể bị giới hạn chỉ nhìn thấy các giá trị lương nhỏ hơn \$30,000. Tương tự như vậy, một người quản lý có thể bị hạn chế khi chỉ nhìn thấy các giá trị lương của các nhân viên trong bộ phận của mình quản lý. Có hai loại kiểm soát truy nhập phụ thuộc dữ liệu là: *kiểm soát truy nhập dựa trên khung nhìn (view-based access control)* và *sửa đổi truy vấn (query modification)*.

Cơ chế *sửa đổi truy vấn* được thực hiện trong các DBMS kiểu Ingres. Cơ chế kiểm soát dựa trên khung nhìn là cơ chế bảo vệ bên dưới của các DBMS dựa trên System R (Griffiths và Wade, 1976). Trong cơ chế *dựa trên khung nhìn*, thay vì truy nhập trực tiếp vào bảng quan hệ cơ sở, người dùng chỉ được truy nhập vào các bảng khung nhìn ảo.

2.2.2.1.4.1 Kiểm soát truy nhập dựa trên khung nhìn

Một *bảng cơ sở (base relation)* là một quan hệ “thực” trong cơ sở dữ liệu, có nghĩa là nó được lưu trữ trong cơ sở dữ liệu. Một *khung nhìn (View)* là một quan hệ “ảo” được đưa ra từ các bảng cơ sở và các khung nhìn khác. Cơ sở dữ liệu lưu trữ các định nghĩa và tư liệu khung nhìn khi cần thiết.

Để minh họa một khái niệm về khung nhìn, và ứng dụng an toàn của nó, ta xem xét một bảng quan hệ EMPLOYEE được thể hiện trong bảng 2.2. (Giá trị NULL là không có giá trị, thể hiện rằng người dùng Harding không có người quản lý). Câu lệnh SQL sau định nghĩa một khung nhìn có tên là TOY-DEPT.

```
CREATE VIEW TOY-DEPT  
AS      SELECT NAME, SALARY, MANAGER  
FROM    EMPLOYEE WHERE DEPT = 'Toy'
```

NAME	DEPT	SALARY	MANAGER
John	Toy	30000	Harding

Smith	Toy	10,000	Jones
Jones	Toy	15,000	Baker
Baker	Admin	40,000	Harding
Adams	Candy	20,000	Harding
Harding	Admin	50,000	NULL

Bảng 2.2: Cơ sở dữ liệu quan hệ EMPLOYEE

Kết quả của câu lệnh này là một bảng quan hệ ảo (khung nhìn) thể hiện trong bảng 2.3.

NAME	SALARY	MANAGER
Smith	10,000	Jones
Jones	15,000	Baker

Bảng 2.3 Khung nhìn TOY-DEPT

Một người dùng chỉ được phép đọc khung nhìn TOY-DEPT thì không thể thấy nội dung còn lại của bảng, mà chỉ nhìn thấy các bản ghi công nhân ở phòng Toy. Để thấy được tính động của khung nhìn, ta giả sử rằng thêm một nhân viên mới Brown vào bảng EMPLOYEE ban đầu, như trong Bảng 2.4. Khi đó khung nhìn TOY-DEPT sẽ tự động được cập nhật thêm nhân viên Brown, như trong bảng 2.5.

NAM E	DEPT	SALAR Y	MANAG ER
Smith	Toy	10,000	Jones
Jones	Toy	15,000	Baker
Baker	Admin	40,000	Harding

Adams	Candy	20,000	Harding
Harding	Admin	50,000	NULL
Brown	Toy	22,000	Harding

Bảng 2.4 Bảng EMPLOYEE được chỉnh sửa

NAME	SALARY	MANAGER
Smith	10,000	Jones
Jones	15,000	Baker
Brown	22,000	Harding

Bảng 2.5 Khung nhìn TOY-DEPT được tự động chỉnh sửa

Khung nhìn cũng có thể được sử dụng để cung cấp truy nhập vào các thông tin thống kê. Ví dụ khung nhìn dưới đây chỉ ra giá trị lương trung bình của mỗi phòng.

```
CREATE VIEW AVSAL(DEPT, AVG)
AS      SELECT DEPT, AVG(SALARY)
FROM    EMPLOYEE
GROUP BY DEPT
```

Với mục đích truy xuất thông tin, người dùng không cần phân biệt giữa khung nhìn và các bảng cơ sở dữ liệu gốc. Một khung nhìn đơn giản chỉ là một quan hệ khác trong cơ sở dữ liệu, được tự động chỉnh sửa bởi DBMS khi mà bảng cơ sở được chỉnh sửa. Do đó, khung nhìn cung cấp một cơ chế mạnh mẽ cho việc cấp quyền phụ thuộc dữ liệu đối với các yêu cầu đọc dữ liệu. Tuy nhiên có vấn đề xảy ra khi người dùng trực tiếp sửa đổi khung nhìn mà không phải khung nhìn được sửa đổi do bảng cơ sở bị chỉnh sửa. Về mặt lý thuyết, không thể truyền những cập nhật trên khung nhìn vào các bảng cơ sở. Điều này làm giảm sự hữu ích của khung nhìn đối với việc cấp quyền phụ thuộc dữ liệu cho các hoạt động cập nhật.

2.2.2.1.4.2 Sửa đổi truy vấn

Sửa đổi truy vấn là một kỹ thuật khác phục vụ cho việc thực thi kiểm soát truy nhập phụ thuộc dữ liệu đối với nhu cầu truy xuất thông tin. (Nó không được hỗ trợ trong SQL, nhưng được trình bày ở đây cho đầy đủ). Trong kỹ thuật này, mỗi một truy vấn mà người dùng thực hiện sẽ được sửa đổi lại để hạn chế thêm người dùng sao cho phù hợp với quyền của người đó.

Giả sử rằng người quản trị cơ sở dữ liệu đã cấp cho người dùng Thomas khả năng truy vấn vào bảng cơ sở EMPLOYEE để lấy thông tin về các nhân viên thuộc phòng “TOY”, như sau:

```
GRANT    SELECT  
ON        EMPLOYEE  
TO        Thomas  
WHERE    DEPT = 'Toy'
```

Bây giờ giả sử rằng Thomas thực hiện truy vấn sau đây

```
SELECT    NAME, DEPT, SALARY, MANAGER  
FROM      EMPLOYEE
```

Trong trường hợp không có kiểm soát truy nhập thì truy vấn này sẽ trả lại toàn bộ bảng EMPLOYEE. Tuy nhiên, do được cấp truyền GRANT ở trên, nên DBMS sẽ tự động sửa đổi truy vấn này như sau:

```
SELECT    NAME, DEPT, SALARY, MANAGER  
FROM      EMPLOYEE  
WHERE    DEPT = 'Toy'
```

Điều này sẽ hạn chế Thomas chỉ truy vấn được phần dữ liệu được cho phép trong bảng EMPLOYEE mà đã được gán quyền SELECT ở trên.

2.2.2.4.3 Gán và thu hồi quyền truy nhập

Việc gán và thu hồi quyền cho phép người dùng lựa chọn và tự động gán các đặc quyền cho những người dùng khác, và sau đó hủy bỏ chúng nếu muốn. Trong SQL, lệnh gán quyền được thực hiện bằng các câu lệnh GRANT, với các định dạng chung sau đây:

```
GRANT    privileges
```

```
[ON      relation]
TO      users
[WITH   GRANT OPTION]
```

Lệnh GRANT áp dụng đối với các bảng quan hệ cơ sở, cũng như các khung nhìn. Tùy chọn ON và WITH biểu thị rằng đây là những tùy chọn và chúng không nhất thiết phải có mặt trong tất cả các lệnh GRANT.

Ví dụ một vài câu lệnh GRANT như sau:

- GRANT SELECT ON EMPLOYEE TO TOM

Cho phép Tom thực hiện lệnh SELECT trên bảng EMPLOYEE

- GRANT SELECT, UPDATE(SALARY) ON EMPLOYEE TO
TOM

Cho phép Tom chỉnh sửa trường SALARY của bảng EMPLOYEE

- GRANT INSERT, DELETE ON EMPLOYEE TO TOM, DICK,
HARRY

Cho phép Tom, Dick, và Harry chèn hàng mới và xóa những hàng trong bảngEMPLOYEE

- GRANT SELECT ON EMPLOYEE TO TOM WITH GRANT
OPTION

Cho phép Tom thực hiện SELECT trên bảng EMPLOYEE, ngoài ra Tom còn có thể gán quyền SELECT cho người khác.

- GRANT DBA TO JILL WITH GRANT OPTION

Cho phép quyền quản trị CSDL cho Jill. Cho phép Jill hoạt động như một người quản trị với nhiều đặc quyền. Trường hợp này Jill có thể cấp quyền quản trị cho những người dùng khác (với GRANT OPTION).

Thu hồi quyền trong SQL được thực hiện bằng lệnh REVOKE như sau:

```
REVOKE privileges
[ON      relation]
FROM    users
```

Một số ví dụ về thu hồi được đưa ra dưới đây. Ý nghĩa của thu hồi phụ thuộc vào người thực hiện nó. Do đó các ví dụ được đưa ra như một chuỗi các hành động. Từng bước sẽ xác định người nào thực hiện lệnh REVOKE đó.

DICK: GRANT SELECT ON EMPLOYEE TO TOM

DICK: REVOKE SELECT ON EMPLOYEE FROM TOM

Dick gán quyền cho Tom và sau đó thu hồi quyền này.

DICK: GRANT SELECT ON EMPLOYEE TO TOM

HARRY: GRANT SELECT ON EMPLOYEE TO TOM

DICK: REVOKE SELECT ON EMPLOYEE FROM TOM

Dick thu hồi quyền SELECT của Tom. Tuy nhiên Tom có quyền SELECT được cấp phép từ Harry

DICK: GRANT SELECT ON EMPLOYEE TO JOE WITH GRANT OPTION

JOE: GRANT SELECT ON EMPLOYEE TO TOM

DICK: REVOKE SELECT ON EMPLOYEE FROM JOE

Lệnh thu hồi này sẽ thu hồi quyền SELECT mà Dick gán cho Joe, và cũng gián tiếp thu hồi quyền SELECT mà Joe gán cho Tom. Đây được gọi là thu hồi quyền đê quy (cascading revocation).

DICK: GRANT SELECT ON EMPLOYEE TO JOE WITH GRANT OPTION

HARRY: GRANT SELECT ON EMPLOYEE TO JOE WITH GRANT OPTION

JOE: GRANT SELECT ON EMPLOYEE TO TOM

DICK: REVOKE SELECT ON EMPLOYEE FROM JOE

Thu hồi này của Dick được thực hiện, nhưng Joe và Tom vẫn còn lại quyền SELECT từ Harry.

Tiếp theo, chúng ta sẽ thấy sự khác biệt với các câu lệnh gán và thu hồi sau:

DICK: GRANT SELECT ON EMPLOYEE TO JOE WITH GRANT OPTION

JOE: GRANT SELECT ON EMPLOYEE TO TOM

HARRY: GRANT SELECT ON EMPLOYEE TO JOE WITH GRANT OPTION

DICK: REVOKE SELECT ON EMPLOYEE FROM JOE

Sau khi thực hiện, Joe vẫn còn quyền SELECT (with GRANT OPTION) do Harry cấp. nhưng Tom lại mất quyền SELECT do nhận quyền từ Joe, việc này xảy ra trước khi Harry gán cho Joe, nên Tom bị thu hồi đệ quy.

DICK: GRANT UPDATE(SALARY,DEPT) ON EMPLOYEE TO JOE

DICK: REVOKE UPDATE ON EMPLOYEE FROM JOE

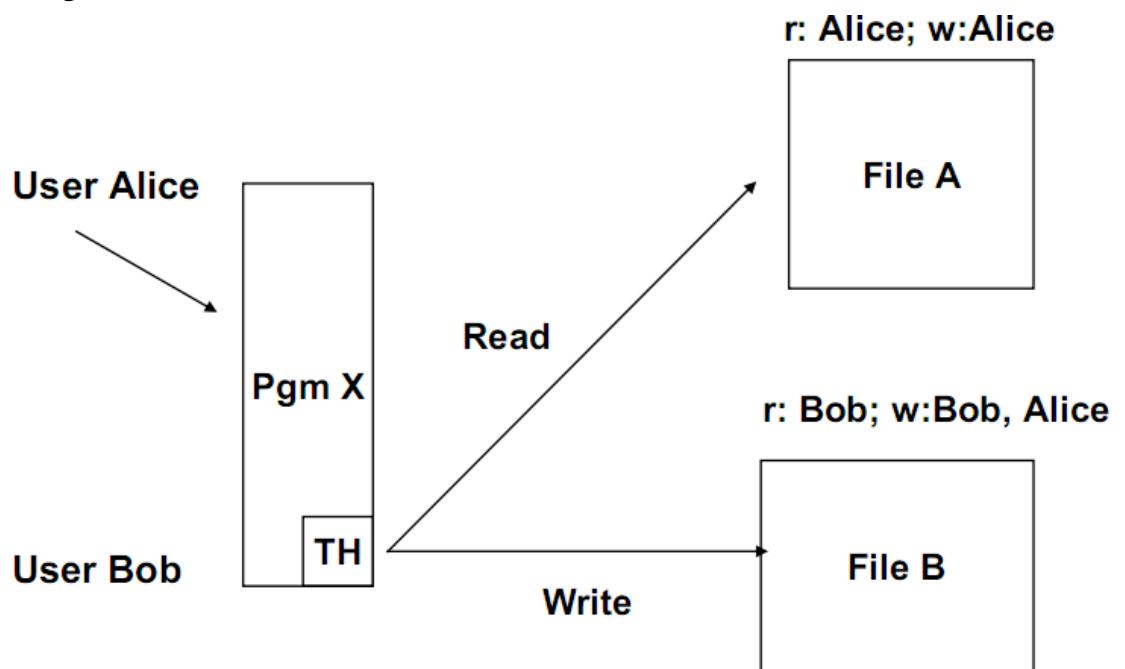
Việc thu hồi này sẽ khiến Joe mất quyền UPDATE ở cả hai cột SALARY và DEPT. Không thể thực hiện thu hồi quyền UPDATE chỉ một trong 2 cột trên.

2.2.2.2 Hạn chế của kiểm soát truy nhập tùy ý

- *Vấn đề cập nhật (Update Problem)*: Cơ chế bảo vệ dựa trên khung nhìn có thuận lợi ở chỗ là nó rất linh hoạt trong việc hỗ trợ các chủ thẻ với quyền truy nhập khác nhau sẽ được đọc dữ liệu từ các khung nhìn khác nhau. Các khung nhìn này được lọc tự động để loại đi những dữ liệu từ bảng cơ sở mà chủ thẻ không được phép truy nhập. Tuy nhiên, bất lợi lại ở chỗ, mỗi khung nhìn chứa một phần dữ liệu của bảng gốc, cho nên khi cập nhật dữ liệu qua khung nhìn, vấn đề toàn vẹn dữ liệu có thể không được đảm bảo đối với những dữ liệu không được chứa trong khung nhìn đó.
- *Cấp quyền đệ quy (Cascading Authorization)*: Nếu hai hay nhiều chủ thẻ có đặc quyền gán hoặc thu hồi quyền với các chủ thẻ khác thì có thể dẫn đến các chuỗi thu hồi đệ quy. Giả sử xét các chủ thẻ s_1, s_2, s_3 và quy tắc truy nhập (s_1, o, t, p) . Chủ thẻ s_2 nhận được quyền (o, t, p) của s_1 . Sau đó, s_2 gán quyền này cho s_3 . Tiếp theo, s_1 lại gán (o, t, p) cho s_3 lần nữa. Cuối cùng, s_2 thu hồi quyền (o, t, p) của s_3 vì một số lý do. Kết quả của các hành động này là s_3 vẫn còn quyền (từ s_1) để truy nhập vào đối tượng o với đặc quyền t và tân từ p . Như vậy, s_2 mặc dù đã thu hồi quyền của s_3 nhưng không biết rằng thực tế s_3 vẫn còn quyền (o, t, p) . Như vậy, việc gán và thu hồi quyền DAC cũng khá phức tạp, nếu không được quản lý tốt thì rất

dễ bị lộ thông tin bí mật. Ví dụ về việc cấp quyền để quy này như được chỉ ra ở phần trước.

- *Tấn công Trojan Horse (Trojan Horse Attacks):* Chính sách kiểm soát truy nhập tùy ý - DAC cho phép đọc thông tin từ một đối tượng và chuyển đến một đối tượng khác; thậm chí nếu một người sử dụng tin cậy không làm điều đó một cách thận trọng thì vẫn có thể tạo ra sơ hở để cho tấn công con ngựa thành Tora sao chép thông tin từ một đối tượng đến một đối tượng khác. Điều này xảy ra, giả sử một chủ thẻ A, đọc đối tượng O₁ sau đó sao chép nội dung của O₁ và ghi vào đối tượng O₂, sau đó A trao quyền đọc hoặc ghi O₂ cho chủ thẻ B. Khi đó B có thể đọc nội dung của O₁ mặc dù chưa được phép. Rõ ràng điều này có thể gây ra mất thông tin nhạy cảm từ một đối tượng.



Hình 2.10 Tấn công Trojan Horse đối với DAC

Để xem xét hạn chế này của kiểm soát truy nhập tùy ý, ta xét ví dụ sau:

TOM: GRANT SELECT ON EMPLOYEE TO DICK

Tom chỉ cấp quyền SELECT cho Dick mà không có thêm GRANT OPTION để không cho Dick được gán quyền SELECT này cho những người

dùng khác. Tuy nhiên điều này dễ dàng bị phá vỡ như sau. Dick tạo ra một bảng mới, là bản sao của EMPLOYEE, nội dung giống như EMPLOYEE, tạm gọi là bảng COPY-EMPLOYEE. Do đó, Dick có mọi quyền trên bảng copy này vì Dick là chủ sở hữu, là người tạo ra bảng đó. Tiếp theo, Dick có thể gán quyền cho Harry như sau:

DICK: GRANT SELECT ON COPY-OF-EMPLOYEE TO HARRY

Lúc này, Harry có thể truy nhập được mọi thông tin từ bảng EMPLOYEE thông qua bảng COPY-EMPLOYEE. Nếu như Dick luôn cập nhật những nội dung của bảng EMPLOYEE vào bảng COPY-EMPLOYEE thì đương nhiên Harry cũng luôn đọc được mọi thông tin cập nhật trên EMPLOYEE.

Trong một trường hợp khác, thì tình hình có thể tệ hơn rất nhiều. Trong ví dụ trên thì Dick được nhắc đến như một người hợp tác vào quá trình này. Bay giờ giả sử Dick là một người đáng tin cậy của Tom và sẽ không có ý định phá hoại hoặc làm sai lệch quyền mà Tom cấp phép trên bảng EMPLOYEE. Tuy nhiên, Dick sử dụng một trình soạn thảo văn bản do Harry cung cấp. Trình soạn thảo này cung cấp mọi dịch vụ soạn thảo mà Dick cần.Thêm vào đó, Harry đã lập trình để nó tạo ra một bản copy của EMPLOYEE là bảng COPY-EMPLOYEE và thực hiện lệnh GRANT ở trên. Phần mềm này được coi như một Trojan Horse, vì ngoài chức năng sử dụng bình thường nó còn có các hành động gian lận nhằm phá vỡ tính an toàn của CSDL.

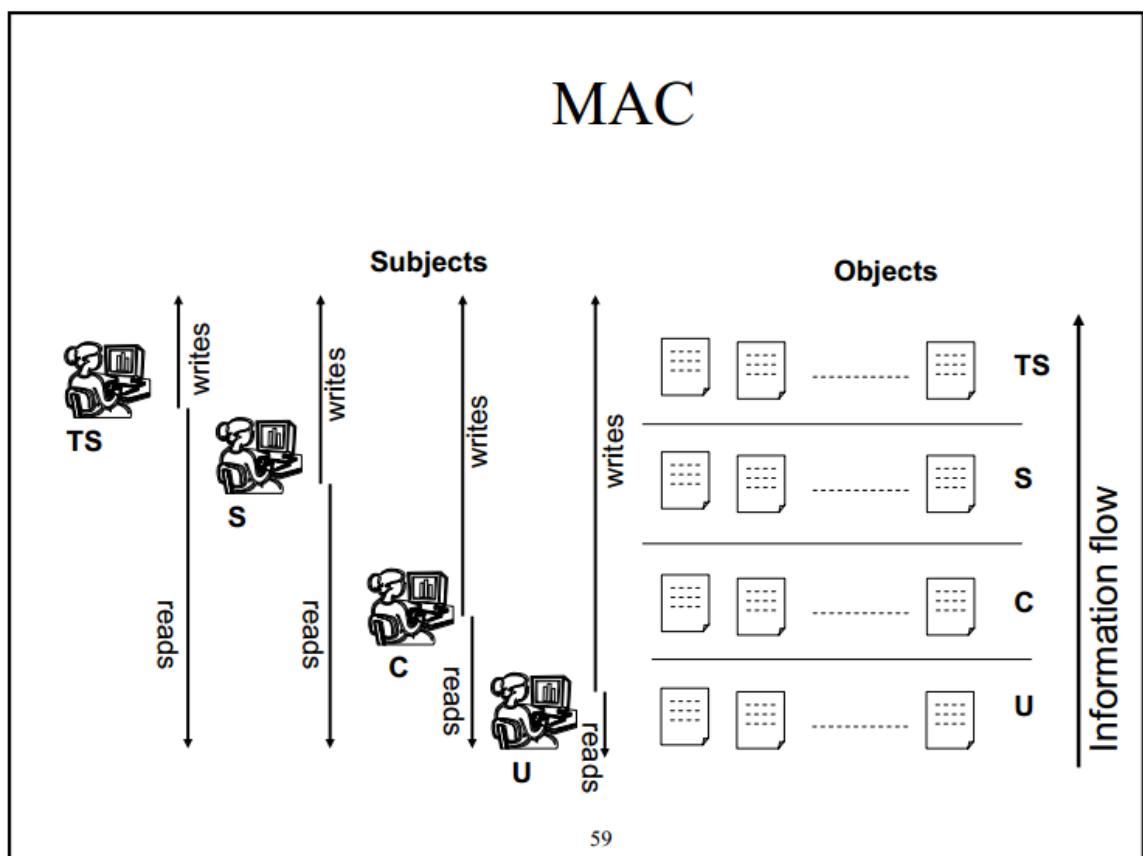
Tóm lại ngay cả khi người sử dụng là tin cậy không cố ý vi phạm, nhưng chúng ta phải đối mặt với những trường hợp như Trojan Horse đã được lập trình trước. Chúng ta có thể yêu cầu các phần mềm chạy trên hệ thống không có sơ hở cho các Trojan Horse, tuy nhiên điều này hầu như không thể thực hiện. Bởi vì, để tìm kiếm một hệ thống hoàn toàn không có mã độc là không thực tế. Giải pháp đặt ra là áp dụng chính sách kiểm soát truy nhập bắt buộc sẽ không thể bị xâm phạm, ngay cả với các Trojan Horse.

2.2.3 Các mô hình an toàn bắt buộc

Trong phần trước, chúng ta đã tìm hiểu về mô hình an toàn tùy ý (discretionary), trong đó việc cấp quyền truy nhập do người dùng kiểm soát.

Người dùng có một đặc quyền với tùy chọn GRANT OPTION có thể cấp quyền cho bất cứ ai. Điều này có rất nhiều hạn chế về vấn đề bảo mật. Trong phần này, chúng ta sẽ tìm hiểu điều khiển truy nhập bắt buộc (mandatory) để khắc phục hạn chế này.

Chính sách an toàn bắt buộc sẽ đảm bảo an toàn cho hệ thống ở mức độ cao hơn so với chính sách an toàn tùy ý, bởi vì ngoài việc kiểm soát quyền truy nhập vào dữ liệu thì chính sách an toàn bắt buộc còn kiểm soát luồng dữ liệu. Hơn thế nữa chính sách an toàn bắt buộc còn khắc phục được các hạn chế về cấu trúc bảo vệ của DAC đã nêu ở trên.



Hình 2.11 Luồng thông tin trong MAC

Chính sách bắt buộc trong kiểm soát truy nhập được áp dụng cho các thông tin có yêu cầu bảo vệ nghiêm ngặt, trong các môi trường mà ở đó dữ liệu hệ thống có thể được phân loại và người dùng cũng được phân loại rõ ràng.

Chính sách bắt buộc cũng có thể được định nghĩa như là một chính sách kiểm soát luồng, bởi vì nó ngăn chặn dòng thông tin đi vào các đối tượng có mức phân loại thấp hơn.

Chính sách bắt buộc quyết định truy nhập vào dữ liệu, thông qua việc định nghĩa các lớp an toàn của *chủ thẻ* và *đối tượng*. Mọi chủ thẻ và đối tượng trong hệ thống đều được gắn với một lớp an toàn (lớp truy nhập)

Một số mô hình an toàn bắt buộc: mô hình Bell – Lapadula (1973, 1974, 1975), mô hình Biba (1977), mô hình Sea View (Denning, 1987), mô hình Dion (1981),...

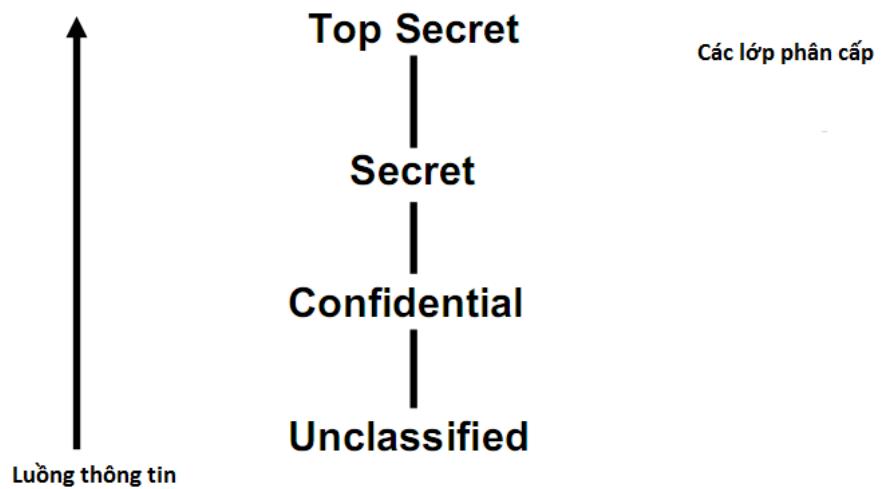
2.2.3.1 Kiểm soát truy nhập bắt buộc

Trong khi mô hình an toàn tùy ý quan tâm đến việc xác định, mô hình hóa và thực thi các quyền truy nhập vào thông tin thì mô hình an toàn bắt buộc lại quan tâm đến lưu lượng của hệ thống.

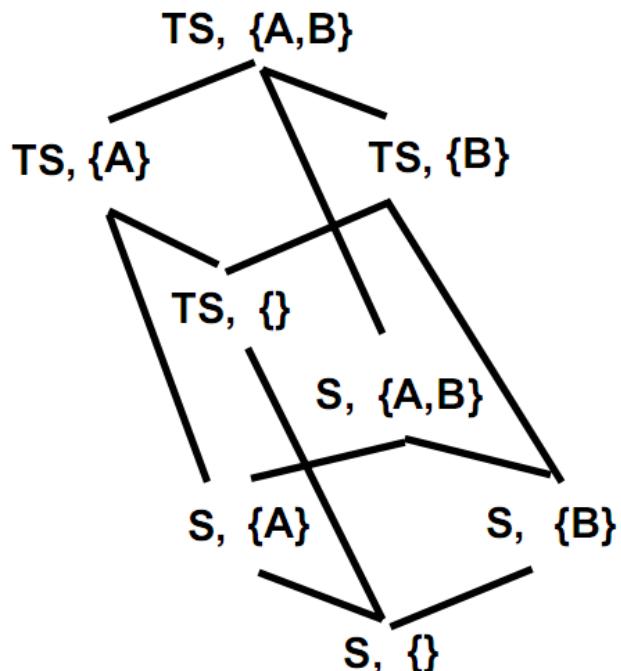
Kiểm soát truy nhập bắt buộc yêu cầu các đối tượng và chủ thẻ được gán các mức an toàn nhất định thông qua các nhãn (label). Nhãn cho một đối tượng o còn được gọi là *phân loại* (*classification*) của đối tượng (*class (o)*) và nhãn cho chủ thẻ s còn được gọi là *mức độ mật* - *clearance* (*clear(s)*). Class đại diện cho sự nhạy cảm của các dữ liệu, còn clearance đại diện cho sự tin cậy của chủ thẻ. Một nhãn an toàn gồm 2 thành phần: một mức độ nhạy cảm của thông tin (dựa theo một danh sách phân cấp mức độ có trước) hoặc các lớp truy nhập (ví dụ: top_secret > secret > confidential > unclassified) và thành phần thứ hai gọi là hạng mục, là một thành phần thuộc tập các kiểu, loại (category) không phân cấp (một tập các lớp của các kiểu đối tượng).

Clearance và class được sắp xếp theo thứ tự cao thấp, kết quả là các nhãn an toàn chỉ được sắp thứ tự một phần, bởi vì thành phần thứ hai của nó không phân cấp. Tập các nhãn an toàn này tạo thành một lưới (lattice). Trong lưới này, thì lớp an toàn c_1 là trội hơn (\supseteq) c_2 nếu mức nhạy cảm của c_1 lớn hơn hoặc bằng mức nhạy cảm của c_2 và các hạng mục (category) trong c_1 chứa các hạng mục (category) trong c_2 . Mô hình an toàn bắt buộc thường được sử dụng

trong quân sự, tuy nhiên cũng được dùng ở các công ty, tổ chức có yêu cầu bảo mật cao.



Hình 2.12 Cấu trúc lưới của các nhãn an toàn



Hình 2.13 Các lớp phân cấp của nhãn an toàn

Trong một hệ thống máy tính, mỗi chương trình chạy bởi một người dùng sẽ thừa hưởng mức độ bảo mật của người dùng đó. Hiểu một cách khác, mức độ bảo mật của người dùng không chỉ áp dụng cho bản thân người dùng mà

còn áp dụng cho các chương trình thực thi bởi người dùng đó. Ví dụ như một chương trình soạn thảo văn bản (text editor) khi được thực thi bởi một người dùng có mức bảo mật Secret thì đó là tiến trình bảo mật mức Secret, nhưng khi chương trình này chạy bởi người dùng mức bảo mật Unclassified, thì tiến trình cũng là Unclassified. Hơn nữa, mức phân loại (class) và mức độ mật (clearance) một khi đã được gán cho người dùng, thì người dùng không thể thay đổi được (trừ người nhân viên an toàn).

Nhắc lại rằng, nhãn an toàn trong quân đội và chính phủ bao gồm hai thành phần là: mức nhạy cảm – có phân cấp và hạng mục – không phân cấp. Các mức nhạy cảm như sau (được liệt kê theo mức độ nhạy cảm giảm dần):

- Top secret (TS)
- Secret (S)
- Confidential (C)
- Unclassified (U)

Các hạng mục bao gồm các mục như: NUCLEAR (Vũ khí hạt nhân), CONVENTIONAL (vũ khí thường), NAVY (hải quân), NATO, ... Nhãn X được gọi là trội hơn nhãn Y nếu X có mức nhạy cảm cao hơn hoặc bằng Y và tập các hạng mục của X chứa tất cả các hạng mục của Y, ví dụ:

- X = (TOP-SECRET, {NUCLEAR, ARMY}) trội hơn Y = (SECRET, {ARMY})
- X = (SECRET, {NUCLEAR, ARMY}) trội hơn Y = (SECRET, {NUCLEAR})
- X = (TOP-SECRET, {NUCLEAR}) không trội hơn Y = (SECRET, {ARMY}), không cái nào trội hơn cái nào.

Lưu ý rằng, hai nhãn thống trị lẫn nhau khi giống hệt nhau.

Các tổ chức thương mại cũng sử dụng nhãn tương tự để bảo vệ thông tin nhạy cảm. Điểm khác biệt chính là các thủ tục gán mức độ mật (clearances) cho người dùng đơn giản hơn nhiều so với trong quân sự và chính phủ.

Trong công nghiệp, thương mại, có thể có các mức nhạy cảm và các hạng mục sau:

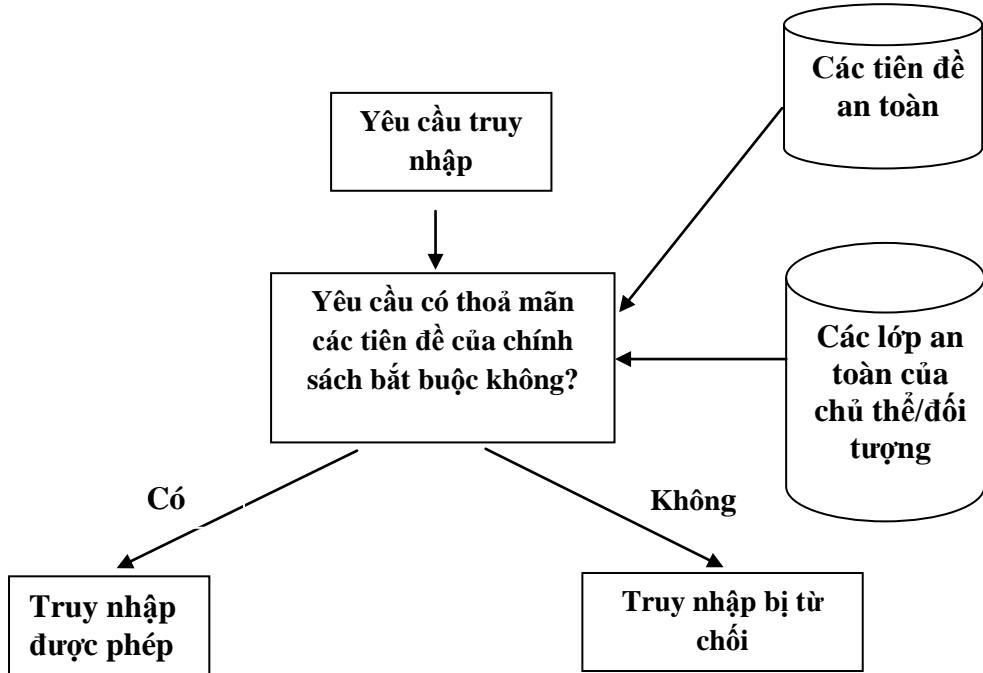
- High Sensitive > Sensitive > Confidential > Unclassified
- Sản lượng – Cá nhân – Kỹ thuật – Quản trị

Hay: Phòng kế hoạch – phòng Marketing – Phòng quản trị

Khi một người dùng đăng nhập vào hệ thống, tài khoản người dùng đó sẽ xác định mức độ bảo mật của phiên làm việc. Mức độ bảo mật của phiên phải được chi phối bởi mức độ mật (clearance) của người dùng đó. Một người dùng mức Secret có thể đăng nhập như mức Unclassified nhưng một người dùng mức Unclassified không thể đăng nhập như là người dùng Secret được.

Để một chủ thẻ được phép truy nhập vào một đối tượng, thì lớp an toàn (nhân) của chủ thẻ phải thỏa mãn các tiêu đề an toàn trong chính sách an toàn của hệ thống.

Người dùng không thể thay đổi các quyền đã được gán, mọi thay đổi chỉ được phép khi có sự đồng ý của người trao quyền (người quản trị trung tâm). Điều này có nghĩa là, người trao quyền sẽ kiểm soát toàn bộ hệ thống trao quyền. Kiểm soát truy nhập thông qua các chính sách bắt buộc được minh họa trong hình dưới đây.



Hình 2.14 Kiểm soát truy nhập bắt buộc

2.2.3.1.1 Mô hình Bell-Lapadula

Các yêu cầu kiểm soát truy nhập bắt buộc (MAC) thường được phát biểu dựa trên mô hình Bell- LaPadula(1976). Một số mô tả về mô hình Bell- Lapadula như sau:

- Xuất hiện năm 1975, do quân đội Mỹ đề xuất.
- Phù hợp sử dụng trong các hệ thống của quân đội và chính phủ
- **Mục đích:** đảm bảo tính bí mật
- Đây là mô hình chính tắc đầu tiên về điều khiển luồng thông tin
- Là một mô hình tĩnh: mức an toàn (nhân an toàn) không thay đổi

Mô hình này bao gồm hai quy tắc. Đầu tiên (thuộc tính đơn giản) bảo vệ các thông tin của cơ sở dữ liệu khỏi việc tiết lộ trái phép, và thứ hai (thuộc tính sao) bảo vệ thông tin khỏi sửa đổi trái phép thông qua việc hạn chế dòng thông tin từ cao xuống thấp.

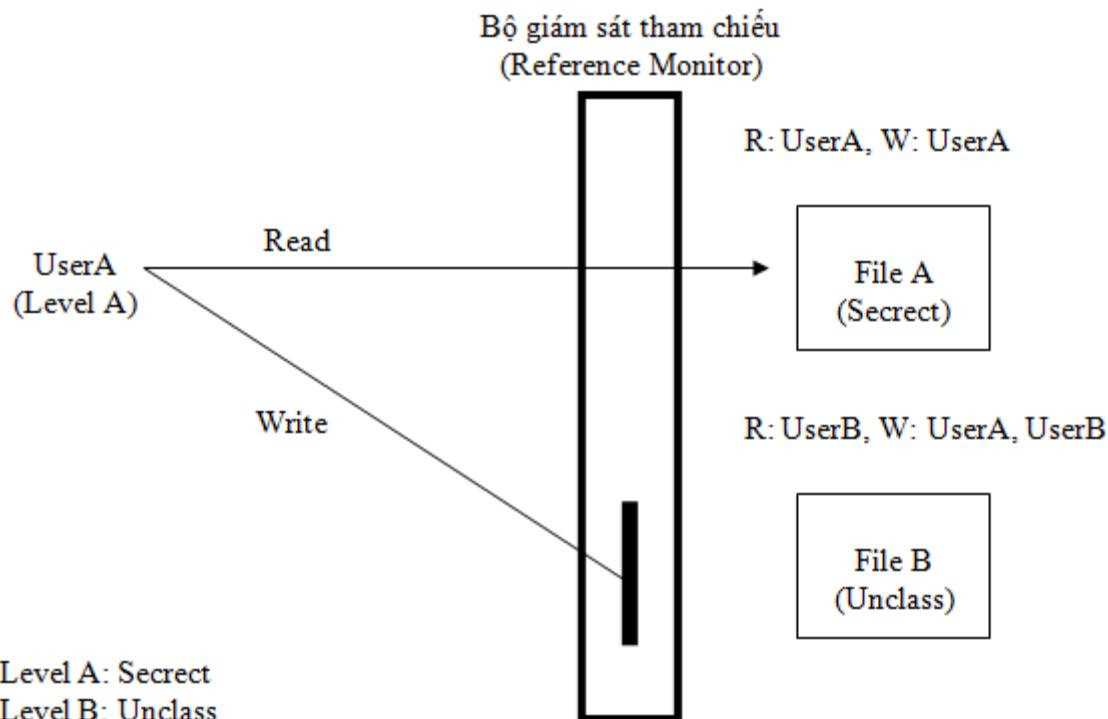
- Thuộc tính an toàn đơn giản:

Một chủ thể S được phép truy nhập đọc đến một đối tượng O chỉ khi $\text{Clearance}(S) \geq \text{Class}(O)$

- Thuộc tính *:

Một chủ thể S được phép truy nhập ghi lên một đối tượng O chỉ khi $\text{Clearance}(S) \leq \text{Class}(O)$.

Hai thuộc tính này đảm bảo rằng không có luồng thông tin trực tiếp nào từ các chủ thể mức cao xuống các chủ thể mức thấp



Hình 2.15 Mô hình Bell – Lapadula chống được tấn công Trojan Horse

Ví dụ, một đối tượng Secret có thể đọc dữ liệu có nhãn Secret và Unclassified, nhưng không thể ghi xuống dữ liệu Unclassified. Vậy kể cả trường hợp Trojan Horse trong phần mềm, cũng không thể nào copy dữ liệu nhạy cảm từ Secret xuống Unclassified.

2.2.3.2 Mô hình an toàn dữ liệu quan hệ đa mức

Tại mục này, chúng ta sẽ định nghĩa các thành phần cơ bản của mô hình dữ liệu quan hệ an toàn đa mức (*MLS – Multilevel Security*). Trọng tâm của an toàn đa mức là tính bí mật. Cần lưu ý rằng mặc dù an toàn đa cấp được

phát triển chủ yếu cho lĩnh vực quân sự, nhưng nó cũng có thể được áp dụng trong lĩnh vực thương mại. Chúng ta xem xét trường hợp chung nhất trong việc gán nhãn an toàn cho chủ thể và đối tượng.

Các hệ thống đa mức đã thành công, do chúng được xây dựng trên các mô hình an toàn được nghiên cứu đầy đủ về mặt lý thuyết.

CSDL đa mức: CSDL trong các cơ quan chính phủ có các kiểu thông tin thiết yếu, được gọi là các CSDL đa mức. Trong đó, người dùng và dữ liệu được phân thành các mức an toàn khác nhau (chẳng hạn như không phân lớp - U, mật - C, tuyệt mật - S, tối mật - TS). Chủ thể khi truy nhập bị giới hạn bởi những điều khiển truy nhập bắt buộc, là “no read up, no write down”, theo mô hình của Bell - LaPadula. Tuỳ theo mức nhạy cảm của dữ liệu và người dùng, mà truy nhập được trao cho người dùng hay không.

Chúng ta giả sử có bốn mức nhạy cảm biểu hiện bằng TS, S, Co, U ($TS > S > Co > U$) và chỉ có một hạng mục (category) duy nhất. Trong mỗi lược đồ quan hệ, TC là một thuộc tính bổ sung để chứa mức phân loại an toàn của các bản ghi trong quan hệ đó.

Đa thể hiện (polyinstantiation):

- Đa thể hiện là một kỹ thuật trong CSDL cho phép CSDL có thể chứa nhiều thể hiện của cùng một dữ liệu với các mức nhạy cảm khác nhau.
- Đa thể hiện tồn tại là do có chính sách bắt buộc (MAC)
- Trong các DBMS quan hệ, có thể có nhiều bản ghi khác nhau nhưng có cùng một khóa chính với các mức nhạy cảm khác nhau.
- Đa thể hiện có thể ảnh hưởng tới các quan hệ, các bản ghi hay các phần tử dữ liệu.
- Đa thể hiện xuất hiện vì các chủ thể với các mức phân loại khác nhau được phép thao tác trên cùng một quan hệ với các lớp truy nhập khác nhau.
- Các bản ghi đa thể hiện là các bản ghi với cùng khóa chính nhưng có các lớp user truy nhập khác nhau gắn với các khóa chính đó.

Sau đây, ta xét ba thể hiện của bảng quan hệ Project như ở hình dưới đây.

Title	Subject	Client	TC
Alpha, S	Development, S	A,S	S
Beta, U	Research, S	B,S	S
Celsius, U	Production, U	C,U	U

Hình 2.16a. Projects

Title	Subject	Client	TC
Beta, U	-,U	-,U	U
Celsius, U	Production, U	C,U	U

Hình 2.16b. Projectv

Title	Subject	Client	TC
Alpha, S	Development, S	A,S	S
Beta, U	Research, S	B,S	S
Celsius, U	Production, U	C,U	U
Alpha, U	Production, U	D, U	U

Hình 2.16c. Sự đa thể hiện (polyinstantiation) ở mức bản ghi

Hình 2.16 Các thể hiện của quan hệ MLS “Project”

Hình 2.16a tương ứng với khung nhìn của một chủ thể s với $clear(s)=S$. Do quy tắc đơn giản của BLP(quy tắc truy nhập đọc) người dùng ở mức U sẽ thấy các thể hiện của bảng Project như ở hình 2.16b. Trong trường hợp này, thuộc tính đơn giản của BLP sẽ tự động lọc ra các dữ liệu trội hơn U .

Hoạt động insert với người dùng mức S: Xét một người dùng s với $clear(s)=S$. Người dùng này muốn thực hiện chèn (insert) một bản ghi $\langle \text{Alpha}, \text{Production}, D \rangle$ và quan hệ $Project_U$ ở hình 2.16b. Vì thuộc tính toàn vẹn khóa nên một DBMS quan hệ chuẩn sẽ không cho phép thực hiện hành động này (vì bản ghi của người dùng Alpha đã tồn tại trong cơ sở dữ liệu, trong trường hợp này Alpha là khóa). Tuy nhiên theo quan điểm an toàn thì hành động chèn bản ghi trên phải được cho phép vì, nếu không sẽ có một kẽm ngầm tồn tại, chủ thể s sẽ kết luận rằng, đó có thể là thông tin nhạy cảm bởi vì anh ta không được phép truy nhập – không thực hiện được câu lệnh insert. Kết quả của câu lệnh insert này được thể hiện trong hình 2.16c.

Hoạt động update với người dùng mức U: hoạt động tương tự xảy ra nếu như một người dùng mức U muốn cập nhật bản ghi $\langle \text{Beta}, \text{null}, \text{null} \rangle$ trong $Project_U$ ở hình 2.16b, bằng việc thay thế các giá trị null bởi các mục dữ liệu cụ thể. Đương nhiên hoạt động này được phép, và lại lần nữa dẫn tới đa thể hiện trong bảng Project.

Hoạt động update với người dùng mức S: một ví dụ khác của đa thể hiện (polyinstantiation), khi chủ thể s với $clear(s)=S$ muốn cập nhật bản ghi $\langle \text{Celsius}, \text{Production}, C \rangle$. Lưu ý rằng, bản ghi này được gắn các nhãn trong các trường đều là U ($U < S$). Trong các hệ thống hỗ trợ MAC, một hoạt động cập nhật như vậy sẽ không được phép, do thuộc tính * của mô hình Bell-Lapadula. Điều này là cần thiết để ngăn chặn luồng thông tin từ các chủ thể mức S xuống các chủ thể mức U . Bởi vậy, nếu một chủ thể mức S muốn cập nhật bản ghi đó, thì bản ghi được cập nhật lại là một thể hiện khác của bản ghi ban đầu, một lần nữa lại tạo ra sự đa thể hiện.

Vấn đề đa thể hiện xuất hiện nhằm tránh kẽm ngầm (convert channel). Lampson (1973) đã định nghĩa một kẽm ngầm như một cách để đi vào luồng thông tin. Như trong ví dụ trên, khi một người dùng mức S muốn chèn (insert)

một bản ghi có mức nhạy cảm U vào cơ sở dữ liệu, nếu hoạt động này bị từ chối (vì đã tồn tại một bản ghi như thế ở mức cao hơn) thì người dùng có thể suy diễn ra sự tồn tại của bản ghi này, kết quả là có một kẽm ngầm xảy ra. Vấn đề không chỉ ở việc suy diễn ra một bản ghi ở mức cao, mà hơn thế, thông tin bất kỳ có thể nhìn thấy ở mức cao, đều có thể bị gửi qua một kẽm ngầm tới mức thấp hơn.

Lý thuyết về các mô hình dữ liệu được xây dựng dựa trên các khái niệm, mà trong thực tế chỉ được biểu diễn một lần trong cơ sở dữ liệu. Do sự đa hình nên thuộc tính cơ bản này không còn đúng với các cơ sở dữ liệu MLS, cho nên yêu cầu phát triển một lý thuyết mới. Lý thuyết mới về MLS được đề cao bởi các nhà nghiên cứu của dự án SeaView. Chẳng hạn, Denning và các cộng sự năm 1988 hoặc Lunt và các cộng sự năm 1990. Các cuộc thảo luận sau đó về mô hình dữ liệu quan hệ MLS chủ yếu dựa vào mô hình được phát triển bởi Jajodia và Sandhu năm 1991.

2.2.3.3 Hạn chế của kiểm soát kiểu bắt buộc

Mặc dù bị giới hạn hơn mô hình DAC, nhưng kỹ thuật MAC cần một số mở rộng để có thể ứng dụng được cho các cơ sở dữ liệu một cách hiệu quả. Cụ thể hơn, chúng ta thấy những hạn chế trong cơ sở dữ liệu an toàn đa mức và các kiểm soát truy nhập bắt buộc dựa trên mô hình BLP như sau:

Độ mịn của đối tượng an toàn:

Độ mịn hay mức chia nhỏ của dữ liệu được gán nhãn vẫn chưa được thống nhất. Việc bảo vệ được sắp xếp từ bảo vệ toàn bộ cơ sở dữ liệu đến bảo vệ tập tin, bảo vệ bảng, đến thuộc tính hoặc thậm chí cả các giá trị thuộc tính. Trong bất cứ trường hợp nào, gán nhãn một cách cẩn thận là điều cần thiết nếu không có thể dẫn đến việc gán nhãn không đầy đủ hoặc không nhất quán.

Thiếu kỹ thuật gán nhãn an toàn tự động

Cơ sở dữ liệu thường xuyên chứa một lượng lớn dữ liệu, phục vụ nhiều người sử dụng và dữ liệu được gán nhãn là không có sẵn trong nhiều ứng dụng thông thường. Đó là lý do việc gán nhãn an toàn bằng tay là cần thiết

nhưng đó lại là một quá trình rất dài, tốn nhiều công sức đối với các cơ sở dữ liệu lớn. Do đó, cần thiết phải hỗ trợ về kỹ thuật, cụ thể là các hướng dẫn và mục tiêu thiết kế cho các cơ sở dữ liệu đa mức, các công cụ giúp quyết định các đối tượng an toàn liên quan và các công cụ gợi ý các mức phân loại (class) và mức độ mật (clearance).

Quy tắc truy nhập N-người

Do chính sách luồng thông tin mà những người dùng mức cao hơn bị hạn chế ghi lên những mục dữ liệu ở mức phân loại thấp hơn. Tuy nhiên, các chính sách của tổ chức có thể cần những nhiệm vụ cụ thể được tiến hành bởi hai hay nhiều người dùng với những mức độ mật khác nhau. Chẳng hạn như, xét các chủ thể s_1, s_2 với $clear(s_1) > clear(s_2)$, mục dữ liệu d với $class(d) = clear(s_2)$. Khi đó, quy tắc của doanh nghiệp yêu cầu là truy nhập ghi của s_2 lên dữ liệu d cần sự chấp thuận của s_1 . Tuy nhiên, quy tắc truy nhập ghi của Bell – Lapadula yêu cầu mức độ mật của s_1 và s_2 phải như nhau. Điều này là không thích hợp cho các ứng dụng doanh nghiệp của công nghệ cơ sở dữ liệu đa mức.

Kênh ngầm (covert channel)

Thật không may điều khiển truy nhập bắt buộc (mandatory access) không giải quyết được hoàn toàn vấn đề Trojan Horse. Một chương trình đang chạy ở mức Secret bị ngăn không được ghi dữ liệu xuống mức Unclassified nếu sử dụng quy tắc của mô hình Bell-Lapadula. Tuy nhiên có một cách khác để trao đổi thông tin với các chương trình mức Unclassified.

Ví dụ, một chương trình Secret có thể giành được một lượng lớn bộ nhớ trong hệ thống. Điều này có thể được phát hiện bởi một chương trình Unclassified vì chương trình này có thể quan sát xem bộ nhớ hiện thời còn bao nhiêu. Kể cả khi chương trình Unclassified bị ngăn chặn không được quan sát trực tiếp dung lượng bộ nhớ sẵn có còn bao nhiêu thì nó có thể làm điều này một cách gián tiếp bằng việc yêu cầu một lượng lớn bộ nhớ dành cho nó. Việc chấp nhận hay từ chối yêu cầu này sẽ hé lộ ra một vài thông tin cho chương trình Unclassified về lượng bộ nhớ còn trống.

Phương pháp truyền thông tin gián tiếp như vậy được gọi là *kênh ngầm* (*covert channel*). Kênh ngầm này thể hiện một vấn đề lớn đối với an toàn đa mức. Chúng rất khó bị phát hiện và khi phát hiện được cũng rất khó để ngăn chặn. Kênh bí mật này có xu hướng gây nhiễu bởi vì nó có khả năng suy diễn hoạt động của những người dùng khác trong hệ thống. Tuy nhiên các kỹ thuật mã chuẩn cho truyền tin trên các kênh nhiễu có thể được sử dụng bởi Trojan horse để đạt được truyền thông không lỗi, với tỷ lệ truyền dữ liệu có thể rất cao như vài triệu bit mỗi giây.

2.2.4 Các mô hình an toàn khác

Ngoài hai mô hình kiểm soát truy nhập chính là MAC và DAC như chúng ta đã tìm hiểu, còn có nhiều mô hình an toàn khác, chẳng hạn như:

Mô hình Biba: là một biến thể của mô hình Bell – Lapadula mà tập trung chính vào việc đảm bảo tính toàn vẹn thông tin trong một hệ thống.

Mô hình Clark-Wilson: nhằm ngăn chặn các người dùng có quyền thực hiện các sửa đổi không được phép trên dữ liệu. Mô hình này thực hiện một hệ thống với bộ ba – một chủ thẻ, một chương trình và một đối tượng.

Mô hình bức tường Trung Hoa (Chinese Wall model): là sự kết hợp giữa thương mại tự do với các điều khiển bắt buộc theo luật. Nó được ứng dụng trong hoạt động của nhiều tổ chức tài chính.

Mô hình mắt lưới (Lattice model): liên quan đến các thông tin quân sự. Các mô hình điều khiển truy nhập dựa trên lưới được phát triển vào đầu những năm 1970 để giải quyết vấn đề đảm bảo tính bí mật của thông tin quân sự.

Chi tiết về các mô hình này, bạn đọc có thể tham khảo trong các tài liệu [6,7,8]

2.3. CÂU HỎI ÔN TẬP

1. Thế nào là mô hình an toàn, và chính sách an toàn?
2. Mô tả quy tắc an toàn và cơ chế an toàn trong các hệ thống cơ sở dữ liệu.

3. Trình bày những đặc điểm chính của mô hình kiểm soát truy nhập bắt buộc (MAC)? Mô hình này được cài đặt trong các DBMS nào?
4. Trình bày những đặc điểm chính của mô hình kiểm soát truy nhập tùy ý (DAC)? Mô hình này được cài đặt trong các DBMS nào?
5. So sánh hai mô hình MAC với DAC.
6. Trình bày những đặc điểm của mô hình System R.
7. Trình bày những hiểu biết về mô hình RBAC.

CHƯƠNG 3. AN TOÀN TRONG DBMS

Chương này mô tả tính thiết kế các kiến trúc DBMS an toàn, sau đó tập trung vào các vấn đề an toàn trong DBMS, giới thiệu một số hệ quản trị cơ sở dữ liệu đang được sử dụng nhiều hiện nay, một số lỗ hổng trong các hệ quản trị. Đồng thời, mô tả các vấn đề an toàn chung trong các hệ quản trị như: xác thực người dùng, cấp quyền, kiểm toán. Cuối chương là phần giới thiệu các cơ chế an toàn trong hệ quản trị cơ sở dữ liệu Oracle.

3.1 THIẾT KẾ DBMS AN TOÀN

CSDL là một tập hợp các dữ liệu được tổ chức và quản lý thông qua phần mềm xác định được gọi là DBMS. Việc đảm bảo an toàn CSDL thông qua các kỹ thuật ở cả hai mức DBMS và OS. Khi thực hiện các yêu cầu an toàn, DBMS có thể khai thác các chức năng an toàn bắt buộc ở mức OS.

Sự khác nhau về an toàn giữa các OS và DBMS được liệt kê sau đây:

- *Độ chi tiết của đối tượng (Object granularity)*: Trong OS, độ chi tiết ở mức tệp (file), thiết bị. Trong DBMS, nó chi tiết hơn (ví dụ như: các quan hệ, các hàng, các cột, các trường).
- *Các tương quan ngữ nghĩa trong dữ liệu (Semantic correlations among data)*: Dữ liệu trong một CSDL có ngữ nghĩa và liên quan với nhau thông qua các quan hệ ngữ nghĩa. Do vậy, nên tuân theo các kiểu kiểm soát truy nhập khác nhau, tùy thuộc vào các nội dung của đối tượng, ngữ cảnh và lược sử truy nhập, để bảo đảm thực hiện chính xác các yêu cầu an toàn trong dữ liệu. Các file trong OS thì không liên quan đến nhau.
- *Siêu dữ liệu (Metadata)*: Siêu dữ liệu tồn tại trong một DBMS, cung cấp thông tin về cấu trúc của dữ liệu trong CSDL, cấu trúc lưu trữ vật lý của các đối tượng CSDL. Siêu dữ liệu thường được lưu giữ trong các từ điển dữ liệu. Ví dụ, trong các CSDL, siêu dữ liệu mô tả các thuộc tính, miền của các thuộc tính, quan hệ giữa các thuộc tính và vị trí phân hoạch CSDL. Trong thực tế, siêu dữ liệu có thể cung cấp các thông tin nhạy cảm về nội dung của CSDL (các kiểu dữ liệu và quan hệ) và có thể

được sử dụng như là một phương pháp nhằm kiểm soát truy nhập vào dữ liệu cơ sở. Không có các mô tả siêu dữ liệu tồn tại trong OS.

- *Các đối tượng logic và vật lý (Logical and physical objects)*: Các đối tượng trong một OS là các đối tượng vật lý (ví dụ: các file, các thiết bị, bộ nhớ và các tiến trình). Các đối tượng trong một DBMS là các đối tượng logic (ví dụ: các quan hệ, các khung nhìn). Các đối tượng logic của DBMS không phụ thuộc vào các đối tượng vật lý của OS, điều này đòi hỏi các yêu cầu và các kỹ thuật an toàn đặc biệt, được định hướng cho việc bảo vệ đối tượng của CSDL.
- *Nhiều loại dữ liệu (Multiple data types)*: Đặc điểm của các CSDL là có rất nhiều kiểu dữ liệu, do đó các CSDL cũng yêu cầu nhiều chế độ truy nhập (ví như chế độ thống kê, chế độ quản trị). Tại mức OS chỉ tồn tại truy nhập vật lý, bao gồm các thao tác trên file như: đọc, ghi và thực hiện.
- *Các đối tượng động và tĩnh (Static and dynamic objects)*: Các đối tượng được OS quản lý là các đối tượng tĩnh và tương ứng với các đối tượng thực. Trong các CSDL, các đối tượng có thể được tạo ra động (ví dụ các khung nhìn hay các kết quả trả lời) và không có các đối tượng thực tương ứng. Ví dụ, khung nhìn của các CSDL được tạo ra động, như các quan hệ ảo có nguồn gốc từ các quan hệ cơ sở được lưu giữ thực tế trong CSDL. Nên định nghĩa các yêu cầu bảo vệ xác định nhằm đối phó với các đối tượng động.
- *Các giao tác đa mức (Multilevel transactions)*: Trong một DBMS thường có các giao tác liên quan đến dữ liệu ở các mức an toàn khác nhau (ví dụ: select, insert, update, delete), bởi vì một đối tượng trong CSDL có thể chứa các dữ liệu ở các mức an toàn khác nhau. DBMS phải đảm bảo các giao tác đa mức được thực hiện theo một cách an toàn. Tại mức OS, một đối tượng chỉ có thể chứa dữ liệu ở một mức an toàn, do đó chỉ có các thao tác cơ bản mới được thực hiện (ví dụ, đọc, ghi, thực hiện), liên quan đến dữ liệu ở một mức an toàn.
- *Thời gian tồn tại của dữ liệu (Data life cycle)*: Dữ liệu trong một CSDL có thời gian tồn tại dài và DBMS có thể đảm bảo việc bảo vệ từ đầu đến

cuối trong suốt thời gian tồn tại của dữ liệu. Nhưng dữ liệu trong một hệ điều hành thường không được lưu trữ một cách an toàn.

3.1.1. Các cơ chế an toàn trong các DBMS

An toàn dữ liệu được quan tâm cùng với việc *khám phá* hoặc *sửa đổi trái phép* thông tin, chẳng hạn như chèn thêm các mục dữ liệu, xoá, thay đổi dữ liệu hiện có. Một DBMS đòi hỏi nhiều tính năng nhằm đạt được các yêu cầu an toàn của một hệ thống thông tin, bao gồm:

- Một số chức năng an toàn chủ chốt phải được OS cung cấp (như quản lý I/O, quản lý tài nguyên chia sẻ, quản lý file...)
- DBMS đóng một vai trò trung tâm bởi vì nó xử lý các quan hệ phức tạp trong dữ liệu.
- DBMS xử lý các ràng buộc an toàn cụ thể tại mức ứng dụng và cần có khả năng ngăn chặn các ứng dụng khám phá hoặc làm hư hại dữ liệu.

Các yêu cầu an toàn chính mà một DBMS nên cung cấp liên quan đến các vấn đề sau đây:

- *Các mức độ chi tiết khác nhau của truy nhập* (*Different degrees of granularity of access*): DBMS cần bảo đảm kiểm soát truy nhập với các mức độ chi tiết khác nhau, như kiểm soát truy nhập tới: từng quan hệ, các cột, hàng, hay các mục dữ liệu trong quan hệ.
- *Các chế độ truy nhập khác nhau* (*Different access modes*): Trong các CSDL, bản chất các chế độ truy nhập được đưa ra dưới dạng các lệnh SQL cơ bản (ví dụ: SELECT, INSERT, UPDATE, DELETE).
- *Các kiểu kiểm soát truy nhập khác nhau* (*Different types of access controls*): Các yêu cầu truy nhập có thể được xử lý, bằng cách sử dụng các kiểu kiểm soát khác nhau.
- *Các kiểm soát phụ thuộc tên* (*Name-dependent controls*) dựa vào tên của đối tượng bị truy nhập (ví dụ ma trận truy nhập).
- *Các kiểm soát phụ thuộc dữ liệu* (*Data-dependent controls*) thực hiện truy nhập phụ thuộc vào các nội dung của đối tượng bị truy nhập (ví dụ 1 user chỉ được đọc các bản ghi về công nhân nếu trường Lương < 1000).

- *Các kiểm soát phụ thuộc ngữ cảnh* (*Context-dependent controls*) chấp thuận hoặc từ chối truy nhập phụ thuộc vào giá trị của một số biến hệ thống (ví dụ như: ngày, tháng, thiết bị đầu cuối yêu cầu – vị trí người dùng, thời gian).
- *Các kiểm soát phụ thuộc lược sử* (*History-dependent controls*) quan tâm đến các thông tin về chuỗi câu truy vấn (ví dụ như: các kiểu câu truy vấn, dữ liệu trả lại, profile của người dùng đang yêu cầu, tần suất yêu cầu).
- *Các kiểm soát phụ thuộc kết quả* (*Result-dependent controls*) thực hiện quyết định truy nhập phụ thuộc vào kết quả của các thủ tục kiểm soát hỗ trợ, chúng là các thủ tục được thực hiện tại thời điểm hỏi.

Trong các CSDL, *kiểm soát truy nhập phụ thuộc dữ liệu* thường được thực hiện thông qua các *cơ chế sửa đổi câu truy vấn* (bổ sung thêm một tân từ vào sau mệnh đề WHERE, ví dụ: *SELECT * FROM Employee WHERE salary < 1000*) hoặc cơ chế sửa đổi dựa vào khung nhìn. Khi sử dụng một kỹ thuật sửa đổi câu truy vấn, câu truy vấn ban đầu (do người dùng yêu cầu) bị hạn chế đến mức nào còn tuỳ thuộc vào các quyền của người dùng.

- *Quyền động* (*Dynamic authorization*): DBMS nên hỗ trợ việc sửa đổi các quyền của người dùng trong khi CSDL đang hoạt động. Điều này tương ứng với nguyên tắc đặc quyền tối thiểu, có thể sửa đổi các quyền tuỳ thuộc vào các nhiệm vụ của người dùng (ví dụ sửa quyền user bằng cách thêm hoặc bớt các Role cho user đó).
- *Bảo vệ đa mức* (*Multilevel protection*): Khi được yêu cầu, DBMS nên tuân theo bảo vệ đa mức, thông qua chính sách bắt buộc (MAC). Các kiểm soát truy nhập bắt buộc (*Mandatory access controls*) dựa vào các *nhân an toàn* được gán cho các đối tượng (là các mục dữ liệu) và các chủ thể (là những người dùng). Trong các môi trường quân sự, các nhân an toàn bao gồm: một thành phần phân cấp (*hierarchical component*) và một tập rỗng các loại không phân cấp (có thể có).

Như vậy, DBMS nên cung cấp các kỹ thuật để định nghĩa các nhân an toàn và gán nhân cho các đối tượng và các chủ thể.

- *Các kênh ngầm (Covert channels)*: DBMS không nên để người dùng thu được các thông tin trái phép thông qua các phương pháp truyền thông gián tiếp. Ví dụ về kênh ngầm như: 1 user thay đổi giá trị một đối tượng, và một user khác theo dõi sự thay đổi này để thu được thông tin mà user 1 muốn truyền ngầm cho anh ta, hay kiểu kênh bằng con ngựa Tora.
- *Các kiểm soát suy diễn (Inference controls)*: Các kiểm soát suy diễn nên ngăn chặn người dùng suy diễn dựa vào các thông tin mà họ thu được trong CSDL. Trong một hệ thống CSDL, các vấn đề suy diễn thường liên quan đến việc *gộp* (*aggregation*) và *gắn kết dữ liệu* (*data association*).

DBMS nên cung cấp một cách thức gán các mức phân loại để tập hợp thông tin, phản ánh *mức nhạy cảm* của các mục dữ liệu được gộp. Khi đó, các thông tin (như mối quan hệ của các mục dữ liệu, hoặc tập hợp các mục dữ liệu) sẽ nhạy cảm hơn so với các mục dữ liệu đơn lẻ, nên chúng cần được quản lý một cách phù hợp.

DBMS không nên để người dùng (through qua các kiểm soát suy diễn) biết các thông tin tích luỹ được có mức phân loại ở mức cao, bằng cách sử dụng các mục dữ liệu được phân loại ở mức thấp.

Giải pháp: Trong một CSDL quan hệ có thể có các giải pháp như

- ✓ *Kỹ thuật hạn chế câu truy vấn*: sửa đổi câu truy vấn ban đầu, hoặc huỷ bỏ câu truy vấn.
- ✓ *Kỹ thuật đa thể hiện (polyinstantiation)*
- ✓ *Kiểm toán*.

Cuối cùng, một kiểu suy diễn đặc biệt xảy ra trong các CSDL thống kê, nơi mà người dùng không được phép suy diễn các mục dữ liệu cá nhân từ dữ liệu kiểm toán đã được gộp và đưa ra. Người dùng có thể thu được các dữ liệu này từ các câu truy vấn kiểm toán.

- *Đa thể hiện (polyinstantiation)*: Kỹ thuật này có thể được DBMS sử dụng để ngăn chặn suy diễn, bằng cách cho phép CSDL *có nhiều thể hiện cho cùng một mục dữ liệu*, mỗi thể hiện có một mức phân loại riêng. Trong một CSDL quan hệ có thể có các bộ khác nhau với cùng

một khoá, với mức phân loại khác nhau, ví dụ nếu tồn tại một hàng (được phân loại ở mức cao) và một người dùng (được phân loại ở mức thấp) yêu cầu chèn thêm một hàng mới có cùng khoá. Điều này ngăn chặn người dùng (được phân loại ở mức thấp) suy diễn sự tồn tại của hàng (được phân loại ở mức cao) trong CSDL.

- *Kiểm toán (Auditing)*: Các sự kiện liên quan tới an toàn (xảy ra trong khi hệ thống CSDL đang hoạt động) nên được ghi lại và theo một khuôn dạng có cấu trúc, chẳng hạn như: nhật ký hệ thống, vết kiểm toán. Các vết kiểm toán rất hữu ích cho các phân tích về sau để phát hiện ra các mối đe dọa có thể xảy ra cho CSDL. Thông tin kiểm toán cũng hữu ích cho kiểm soát suy diễn, nhờ đó chúng ta có thể kiểm tra được lược sử của các câu truy vấn do người dùng đưa ra, để quyết định xem có nên trả lời câu truy vấn mới hay không, vì câu truy vấn mới này lại liên quan đến các kết quả của các câu truy vấn trước đó, có thể dẫn đến một vi phạm suy diễn.
- *Các kiểm soát luồng (Flow controls)*: Các kiểm soát luồng kiểm tra đích của đầu ra. Chúng ta có thể có được kiểm soát này thông qua một truy nhập được phép (*authorized access*).
- *Không cửa sau (No back door)*: Truy nhập vào dữ liệu chỉ nên xảy ra thông qua DBMS. Phải đảm bảo không có các đường dẫn truy nhập ẩn.
- *Tính chất đồng dạng của các cơ chế (Uniformity of mechanisms)*: Nên sử dụng các cơ chế chung để hỗ trợ các chính sách khác nhau và tất cả các kiểm soát liên quan tới an toàn (các kiểm soát bí mật và toàn vẹn).
- *Hiệu năng hợp lý (Reasonable performance)*: Các kiểm soát an toàn làm tăng thời gian hoạt động, do đó cần tối thiểu hóa các kiểm soát để đảm bảo hiệu năng của hệ thống nhưng vẫn đảm bảo được tính an toàn.

Các nguyên tắc đảm bảo toàn vẹn thông tin: chúng độc lập với ngữ cảnh của DBMS và các đặc thù của ứng dụng. Lợi ích của các nguyên tắc này là giúp chúng ta đánh giá các chính sách an toàn của một hệ thống thông tin xác định:

- *Các giao tác đúng đắn (Well-formed transactions)*: Nguyên tắc này còn được gọi là *sự thay đổi có ràng buộc*: chỉ được sửa dữ liệu thông qua các giao tác đúng đắn. Độ chính xác của các giao tác này được chứng

thực với một mức độ đảm bảo nào đó. Các giao tác này chuyển sang các cơ chế DBMS để đảm bảo các thuộc tính cho giao tác. DBMS phải đảm bảo rằng các cập nhật phải được thực hiện thông qua các giao tác, lưu ý rằng CSDL phải được đóng gói trong DBMS thông qua OS.

- *Người dùng được xác thực (Authenticated users)*: Theo nguyên tắc này, không cho phép người dùng thực hiện các thay đổi trừ khi định danh của họ được xác thực chính xác. Việc xác thực người dùng thuộc trách nhiệm của OS và không cần phải lặp lại trong DBMS. Việc xác thực người dùng thường do hệ điều hành thực hiện, không cần hệ quản trị phải làm điều đó. Việc xác thực làm cơ sở cho một số nguyên tắc khác được liệt kê sau đây (đặc quyền tối thiểu, tách bạch nhiệm vụ, ủy quyền).
- *Đặc quyền tối thiểu (Least privilege)*: Đây là một nguyên tắc giới hạn người dùng chỉ được làm việc với một tập tối thiểu các đặc quyền và tài nguyên cần thiết để thực hiện nhiệm vụ của mình. Đặc quyền tối thiểu chuyển sang các cơ chế DBMS cho các thao tác "đọc" và "ghi".
- *Tách bạch nhiệm vụ (Separation of duties)*: Nguyên tắc này được đưa ra nhằm hạn chế tối đa một cá nhân bất kỳ có thể phá hoại dữ liệu, để đảm bảo toàn vẹn dữ liệu. Tách bạch nhiệm vụ được gắn liền với các kiểm soát trên các chuỗi giao tác. Hiện tại có nhiều cơ chế nhưng chúng không được thiết kế cho các mục đích toàn vẹn, do vậy gây ra một số bất tiện.
- *Tính liên tục của hoạt động (Continuity of operation)*: Vấn đề này đã nhận được nhiều sự quan tâm, cả về mặt lý thuyết và thực hành, các giải pháp dựa trên cơ sở lặp dữ liệu cũng đã được đề xuất. Đối mặt với các sự cố phá hoại vượt ngoài tầm kiểm soát của tổ chức, nên duy trì các hoạt động của hệ thống sau khi sự cố xảy ra (quan tâm đến các biện pháp an toàn vật lý).
- Việc dựng lại có thể xảy ra ở nhiều mức vết kiểm toán khác nhau, nhiều việc khác nhau như: ghi lại một lược sử đầy đủ về việc sửa đổi giá trị của một *Dựng lại các sự kiện (Reconstruction of events)*: Việc dựng lại các sự kiện trong một DBMS phụ thuộc vào các vết kiểm toán. mục dữ liệu, hoặc lưu giữ nhận dạng của từng cá nhân khi thực hiện từng thay

đối. Một vấn đề đặt ra là chất lượng của dữ liệu trong kết quả kiểm toán cũng phải phù hợp. Một số đề xuất gần đây có sử dụng các kỹ thuật của hệ chuyên gia để lưu giữ và làm sáng tỏ các kết quả kiểm toán.

- *Kiểm tra thực tế (Reality checks):* Việc kiểm tra định kỳ đối với các thực thể thực tế góp phần duy trì các giá trị dữ liệu chính xác trong hệ thống. DBMS có trách nhiệm duy trì một khung nhìn thích hợp bên trong CSDL, làm cơ sở cho các kiểm tra bên ngoài.
- *Dễ dàng sử dụng an toàn (Ease of safe use):* Cách dễ nhất để điều hành một hệ thống cũng phải là cách an toàn nhất. Điều này có nghĩa là các thủ tục an toàn nên đơn giản, phổ biến, dễ sử dụng.
- *Uỷ quyền (Delegation of authority):* Nó quan tâm đến việc gán các đặc quyền cho tổ chức, lấy các chính sách làm cơ sở, yêu cầu các thủ tục gán phải phản ánh các quy tắc của tổ chức và chúng phải mềm dẻo. Trong các cơ chế tùy ý, việc uỷ quyền tuỳ thuộc vào bản thân chủ thể, có thể trao hoặc thu hồi, huỷ bỏ các quyền cho người dùng khác. DBMS nên cung cấp các cơ chế cho việc quản lý thu hồi. Uỷ quyền là một khả năng cần thiết, nhằm phản ánh cấu trúc phân cấp của tổ chức và nên thực hiện tuân theo các quy tắc (đã được định nghĩa trong tổ chức). Uỷ quyền trong một DBMS thường tuân theo các lệnh SQL GRANT/REVOKE.

Nói riêng, khi quan tâm đến tính toàn vẹn của một DBMS, trong Sandhu và Jajodia (1990) các nguyên tắc được phân nhóm như sau:

- *Nhóm 1:* Các giao tác đúng đắn, tính liên tục của hoạt động. Các nguyên tắc này bao trùm hoàn toàn lên các cơ chế của DBMS.
- *Nhóm 2:* Đặc quyền tối thiểu, tách bạch nhiệm vụ, xây dựng lại các biến cố và uỷ quyền. Nhiều cơ chế mới được yêu cầu cho nhóm này và một số giải pháp đầy triển vọng nhằm mở rộng các cơ chế của DBMS cũng được đưa ra.
- *Nhóm 3:* Người dùng được xác thực, kiểm tra thực tế và dễ dàng sử dụng an toàn. Xác thực là trách nhiệm của OS, kiểm tra thực tế tuỳ thuộc vào an toàn tổ chức.

3.1.2 Các kiến trúc của DBMS an toàn

Trong phần này, trình bày một số đặc điểm chính của các kiến trúc DBMS an toàn. Các DBMS an toàn hoạt động theo 2 chế độ là: chế độ “*hệ thống mức cao*” và chế độ “*đa mức*”.

Các DBMS mức an toàn *hệ thống cao*, tất cả những người dùng được chuyển sang mức an toàn cao nhất, trước khi đưa ra dữ liệu và có một người có trách nhiệm xem xét dữ liệu này để đưa ra chúng một cách chính xác. Giải pháp này cho phép người dùng sử dụng các kỹ thuật DBMS hiện có, nhưng phát sinh một số chi phí cho các “thủ tục cho phép” và xem xét dữ liệu thủ công. Chế độ này có thể làm tăng thêm một số rủi ro an toàn khi tất cả những người dùng được chuyển sang mức cho phép cao nhất.

Với chế độ *đa mức*, có thể có nhiều kiểu kiến trúc khác nhau, dựa vào việc sử dụng các DBMS tin cậy và không tin cậy. Các kiến trúc đa mức như: kiến trúc *Trusted Subject* (*chủ thẻ tin cậy*) và các kiến trúc *Woods Hole*, chúng được Woods Hole Summer Study đề xuất năm 1982. Các kiến trúc Woods Hole bao gồm: *Integrity Lock*, *Kernelized* và các kiến trúc *Replicated*. Trong các kiến trúc Trusted Subject, sử dụng cả DBMS tin cậy và OS tin cậy, trong khi đó các kiến trúc Woods Hole chỉ sử dụng DBMS không tin cậy cùng với một bộ lọc tin cậy.

Các kiến trúc DBMS an toàn nhiều mức đã được đề xuất, nhằm đáp ứng các yêu cầu bảo vệ nhiều mức. Một số kiến trúc nhiều mức được đề xuất như: *Integrity Lock*, *Kernelized*, *Replicated*, *Trusted subjects*.

- *Kiến trúc Replicated*: được phát triển ở NRL (Naval Research Laboratory - thư viện nghiên cứu Naval), hiện chưa có hệ thống thương mại nào có kiến trúc này.
- *Kiến trúc Integrity Lock*: được nghiên cứu bởi tập đoàn Mitre, và có trong một sản phẩm thương mại là Trudata.
- *Kiến trúc Kernelized*: nghiên cứu ở SRL (Stanford Research Institute - Viện nghiên cứu Stanford) đã được nhúng vào một phiên bản an toàn của Oracle.

- *Kiến trúc Trusted Subjects*: được phát triển nghiên cứu ASD-TRW. Hầu hết các sản phẩm DBMS thương mại an toàn (như: Sybase, Rubix, Informix, Oracle, DEC) đều thực thi kiến trúc này.

Bảng 3.1, đưa ra một cái nhìn tổng quan về các kiến trúc được sử dụng trong một số DBMS thương mại và trong một số mẫu thử nghiên cứu của DBMS.

<i>Kiến trúc</i>	<i>Các mẫu thử nghiên cứu</i>	<i>DBMS thương mại</i>
Integrity Lock	Mitre	TRUDATA
Kernelized	Sea View	Oracle
Replicated	NRL	-----
Trusted Subject	A1 Secure DBMS (ASD)	Sybase Informix Ingres Oracle DEC Rubix

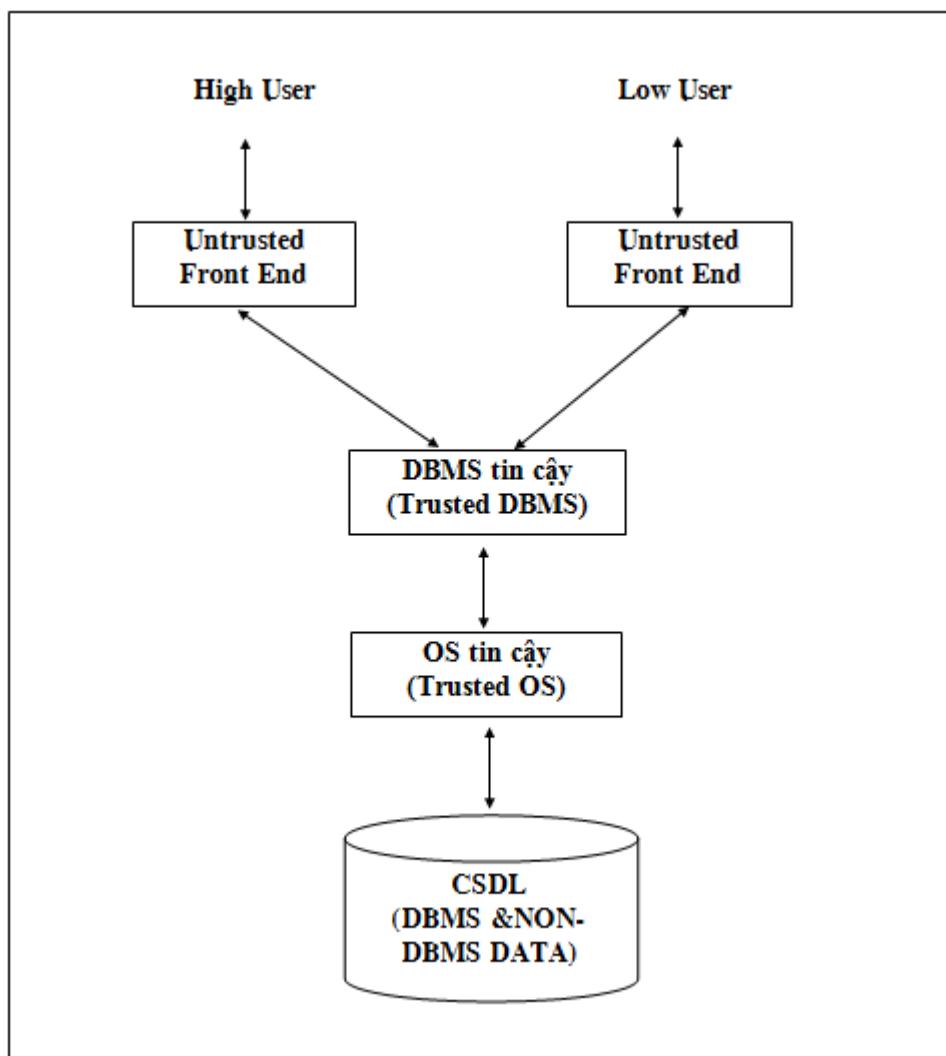
Bảng 3.1 Các kiến trúc mẫu thử DBMS và các sản phẩm thương mại

3.1.2.1 Kiến trúc chủ thẻ tin cậy

Kiến trúc chủ thẻ tin cậy được minh họa trong hình 3.1. Một tập hợp các UFE (*untrusted front end*) được sử dụng để tương tác với người dùng, với các mức cho phép khác nhau (Như đã được trình bày trong hình vẽ, có mức cao và mức thấp). Phần mềm cơ sở dữ liệu này có thể được thực hiện bởi người sử dụng cuối trên chính hệ thống của họ để truy nhập các cơ sở dữ liệu cục bộ nhỏ cũng như kết nối với các cơ sở dữ liệu lớn hơn trên cơ sở dữ liệu Server.

Khi một DBMS tin cậy được sử dụng và hoạt động như là một chủ thẻ tin cậy đối với OS, thì nó cũng được tin cậy, nó thực hiện các truy nhập vật lý vào CSDL. Hoạt động như là một chủ thẻ tin cậy của OS có nghĩa là được miễn một hoặc nhiều khía cạnh nào đó trong chính sách an toàn của OS, nói chung, được miễn các kiểm soát bắt buộc.

DBMS và OS phải được nhìn nhận như là một thực thể, nếu hiểu theo nghĩa thông thường, chúng được ước tính để xác định mức bảo vệ. Trong kiến trúc này, *DBMS có trách nhiệm trong việc bảo vệ đa mức* các đối tượng của CSDL.



Hình 3.1 Kiến trúc chủ thẻ tin cậy

Lưới an toàn được xây dựng theo cách như vậy, định nghĩa mức High, mức Low và một mức DBMS khác với mức High và Low. Nhãn DBMS được gán cho cả các đối tượng và chủ thẻ. Chỉ có các chủ thẻ của DBMS có thể thực hiện các chương trình và truy nhập dữ liệu với một nhãn DBMS. Hơn nữa, các chủ thẻ (có nhãn DBMS) được xem như là các chủ thẻ tin cậy và được miễn các kiểm soát bắt buộc của OS. Theo giải pháp này, có thể nhóm các yếu tố có cùng mức nhạy cảm và lưu giữ chúng trong một đối tượng với mức chi tiết thô, chỉ gán một nhãn cho đối tượng này, hoặc gán nhãn cho từng đối tượng (ví dụ, các bộ, các giá trị). Sybase DBMS tuân theo giải pháp này, với kiến trúc máy khách/máy chủ. Sybase thực hiện gán nhãn mức bản ghi (mức hàng).

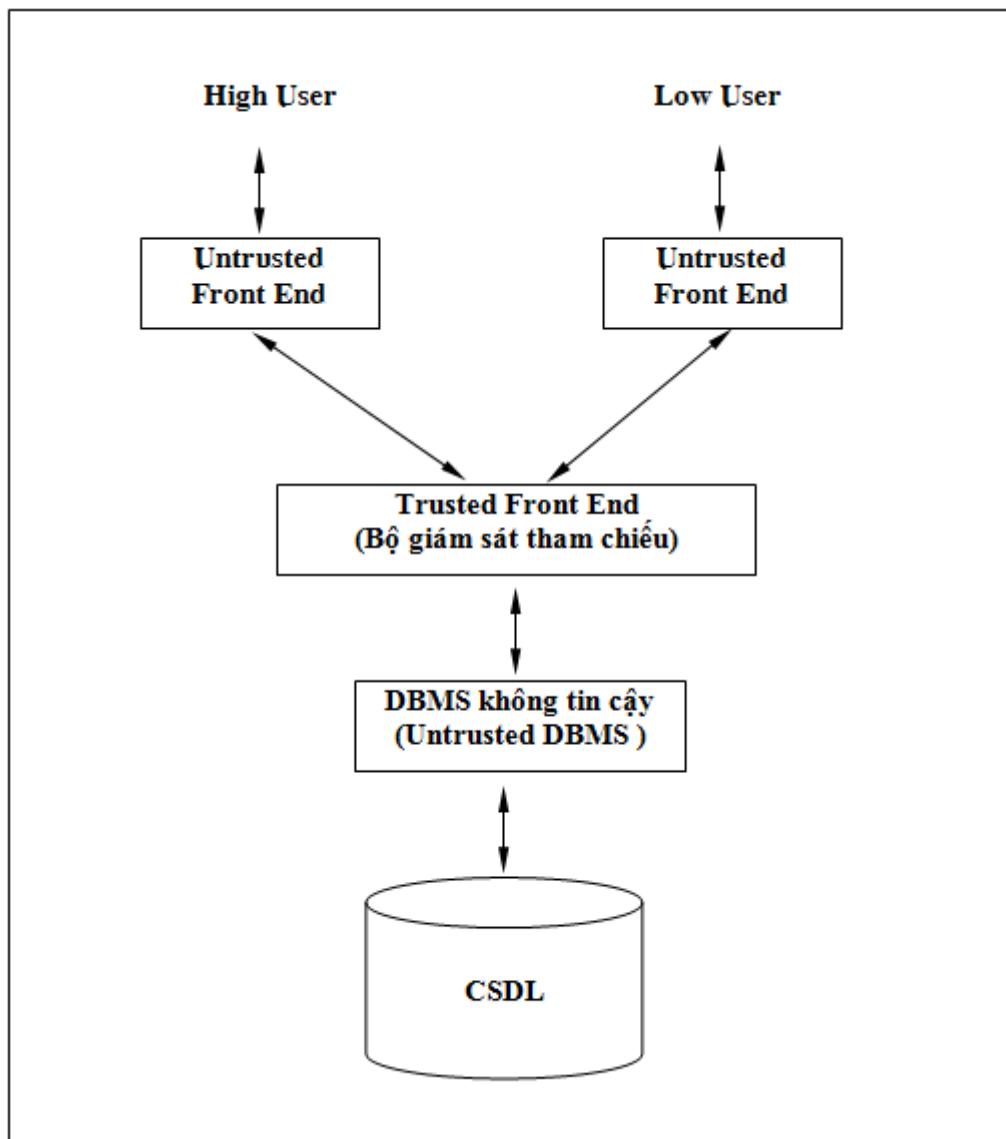
3.1.2.2. Các kiến trúc Woods Hole

- Các kiến trúc Woods Hole sử dụng DBMS không tin cậy cùng với một bộ lọc tin cậy và không quan tâm đến OS có tin cậy hay không.

Các kiến trúc Woods Hole được phân loại như sau:

- ✓ Kiến trúc Integrity Lock
- ✓ Kiến trúc Kernelized
- ✓ Kiến trúc Replicated (còn được gọi là kiến trúc Distributed)

Chúng có thể được miêu tả thông qua một kiến trúc tổng quát, được minh họa trong hình 3.2.



Hình 3.2 Các kiến trúc Woods Hole

Chúng ta nhận thấy rằng một tập hợp các UFE tương tác với những người dùng hoạt động tại các mức cho phép khác nhau, ở đây chúng được đơn giản hóa thành High và Low. Lần lượt, UFE tương tác với một TFE (*trusted front end*), nó hoạt động như một bộ giám sát tham chiếu; Có nghĩa là không thể bỏ qua nó. TFE tương tác với một UBED (*untrusted back end DBMS*), có trách nhiệm trong việc truy nhập dữ liệu vào CSDL. (*Back-end software*: phần mềm này bao gồm phần mềm cơ sở dữ liệu Client/Server và phần mềm mạng chạy trên máy đóng vai trò là Server cơ sở dữ liệu).

Tiếp theo chúng ta mô tả các đặc điểm của từng kiến trúc.

Kiến trúc Integrity Lock (khóa toàn vẹn)

Khoá toàn vẹn được đề xuất lần đầu tiên tại Viện nghiên cứu của Lực lượng Không quân về An toàn cơ sở dữ liệu. Khoá toàn vẹn có thể được dùng để kiểm soát *tính toàn vẹn* và sự *truy nhập* cho cơ sở dữ liệu.

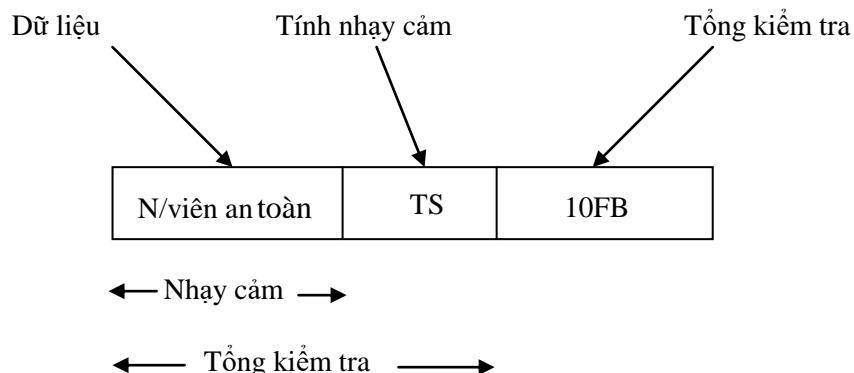
Kiến trúc Integrity lock đã có trong hệ quản trị thương mại TRUDATA.

TRUDATA là một công nghệ cơ sở dữ liệu tin cậy gồm các đặc trưng của các chính sách an toàn nhiều mức (MLS) và bảo đảm cho các sản phẩm DBMS quan hệ hiện có. Công nghệ TRUDATA bao gồm:

- ✓ Một mô hình dữ liệu
- ✓ Một mô hình chính sách an toàn
- ✓ Kiến trúc hệ thống.
- ✓ Định nghĩa một DBMS an toàn nhiều mức tin cậy.

Việc áp dụng công nghệ TRUDATA vào các DBMS hiện có, đảm bảo lớp B1, B2 trong tiêu chuẩn đánh giá DoD.

Một mô hình về khoá toàn vẹn cơ bản được chỉ ra như trên hình vẽ.



Hình 3.3 Khoá toàn vẹn

Như trên hình 3.3, mỗi mục dữ liệu (mỗi ô trong bảng CSDL) chứa ba phần: *bản thân dữ liệu*, *nhãn nhạy cảm* và *tổng kiểm tra*. Nhãn nhạy cảm xác định tính nhạy cảm của dữ liệu, còn tổng kiểm tra được tính qua cả dữ liệu lẫn nhãn nhạy cảm để ngăn sự biến đổi trái phép dữ liệu hoặc nhãn của nó.

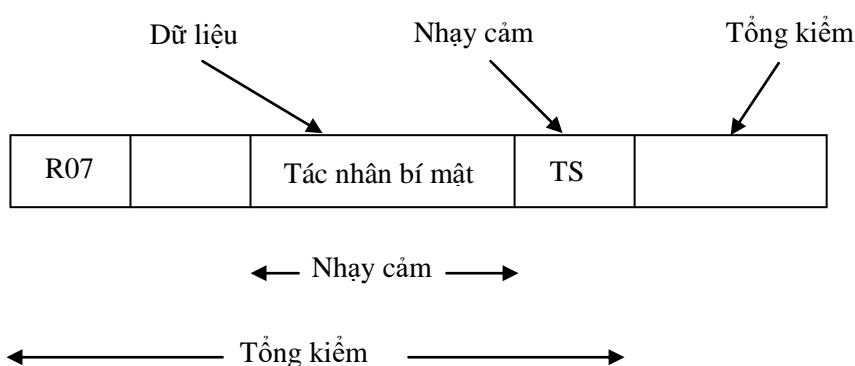
Dữ liệu được lưu ở dạng rõ cho hiệu quả vì DBMS có thể cần đến để đánh giá nhiều trường khi chọn các bản ghi phù hợp với một câu truy vấn. Tổng kiểm tra được mã hóa bằng một khóa toàn vẹn theo thuật toán DES.

Nhận nhạy cảm cần có các tính chất:

- ✓ Không thể giả mạo được để những chủ thể ác ý không thể tạo ra mức nhạy cảm mới cho một phần tử.
- ✓ Duy nhất để kẻ ác ý không thể sao chép mức nhạy cảm từ phần tử khác
- ✓ Được che đậy sao cho kẻ ác ý không thể xác định được mức nhạy cảm của một phần tử tùy ý.

Tổng kiểm tra mật mã: Phần thứ ba của khoá toàn vẹn đối với một trường là một mã phát hiện sai, được gọi là *tổng kiểm tra Mật mã*. Để đảm bảo rằng, giá trị dữ liệu hay nhận nhạy cảm của nó không bị thay đổi, tổng kiểm tra này phải là duy nhất đối với phần tử đã cho (một ô trong bảng) và phải chứa cả dữ liệu của phần tử và một cái gì đó để ràng buộc dữ liệu đó vào một vị trí cụ thể trong cơ sở dữ liệu.

Ví dụ: một tổng kiểm tra mật mã thích hợp gồm thuộc tính duy nhất đối với bản ghi (*số hiệu bản ghi*), thuộc tính duy nhất đối với trường dữ liệu này bên trong bản ghi (*tên thuộc tính trường*), dữ liệu của phần tử này và *nhạy cảm* của phần tử. Bốn thành phần này đảm bảo chống lại sự thay đổi, sao chép hoặc chuyển dữ liệu. Có thể tính tổng kiểm tra bằng một thuật toán mã mạnh, chẳng hạn như DES.



Hình 3.4 Ví dụ tổng kiểm tra mật mã

Lưu ý tổng kiểm tra cần được mã hoá, còn nhãn nhạy cảm không cần mã hóa nhưng sẽ được che giấu đi.

Theo giải pháp này, người dùng được kết nối thông qua các giao diện front end không tin cậy, thực hiện tiền xử lý và hậu xử lý các câu truy vấn. Một *TFE* (*còn được gọi là một bộ lọc tin cậy*) được chèn vào giữa các UFE và DBMS không tin cậy.

TFE có trách nhiệm trong việc tuân theo các chức năng an toàn và bảo vệ nhiều mức, hoạt động như là một TCB (Trusted Computing Base). *TFE* tuân theo bảo vệ nhiều mức bằng cách gắn nhãn an toàn cho các đối tượng của CSDL, theo các dạng tem. Tem là một trường đặc biệt của một đối tượng, lưu giữ các thông tin (liên quan đến nhãn an toàn và các dữ liệu kiểm soát liên quan khác) trong một khuôn dạng đã được mã hoá, được tạo ra bằng cách sử dụng một kỹ thuật niêm phong mật mã (*cryptoseal mechanism*), được gọi là *Integrity Lock*.

TFE tiến hành tạo và phê chuẩn các tem, ngay khi dữ liệu được lưu giữ và ngay khi nhận được dữ liệu từ CSDL. *TFE* sinh ra các tem, bằng cách sử dụng các kỹ thuật tổng kiểm tra (*checksum*) (Nó sử dụng một hoặc nhiều khoá bí mật, chỉ duy nhất *TFE* biết được khoá này), bao quanh dữ liệu và được lưu giữ trong CSDL theo một khuôn dạng đã được mã hoá.

Tại thời điểm nhận được dữ liệu từ CSDL, *TFE* tính toán lại các tem và so khớp với bản được lưu giữ, để phát hiện ra sự sai khớp, trước khi dữ liệu được chuyển cho người dùng.

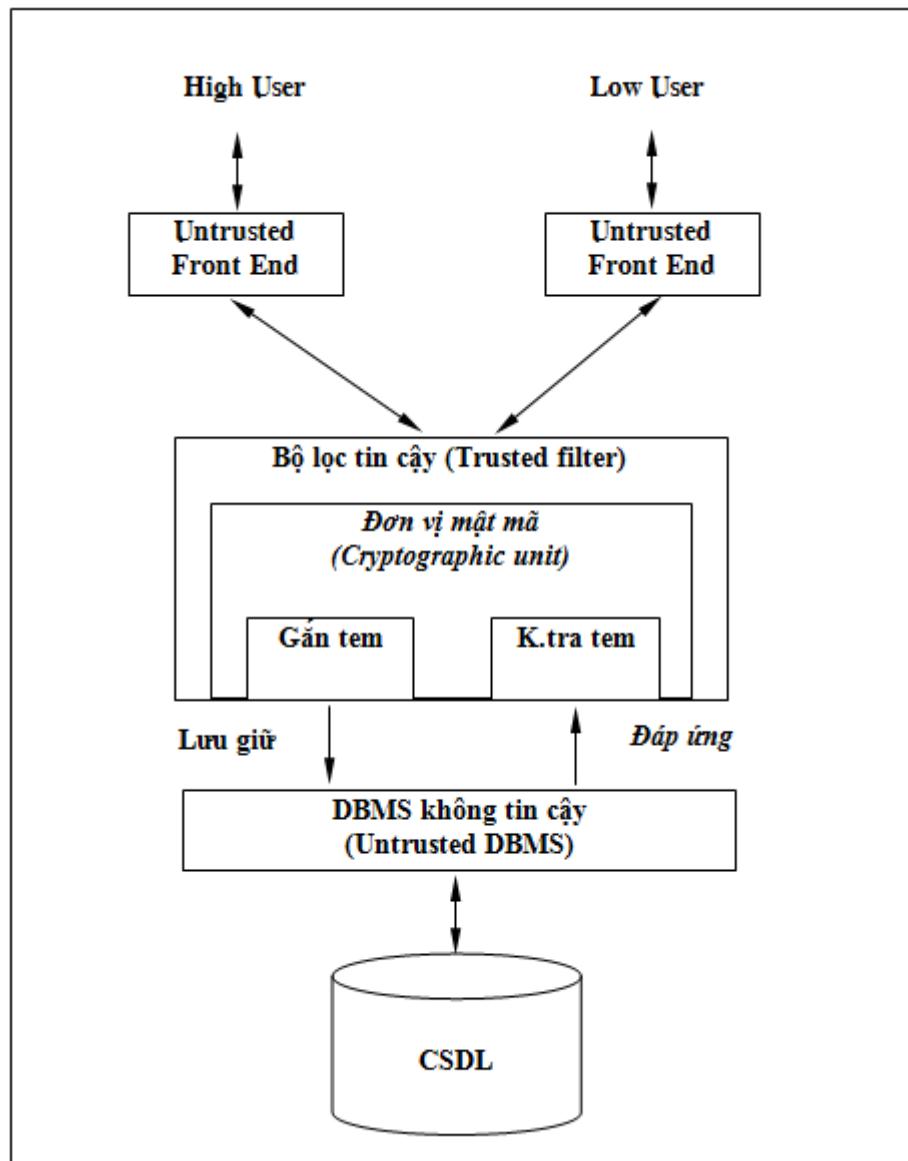
Giả sử khi người dùng muốn insert một mục dữ liệu, *TFE* sẽ tính tổng kiểm tra - tem bằng mức nhạy cảm của người dùng đó và dữ liệu được insert sau đó mã hoá tem này bằng một khoá K, và lưu vào trong CSDL cùng với mục dữ liệu đó (gắn với mục dữ liệu). Khi có người dùng sửa đổi trái phép một mục dữ liệu nào đó, *TFE* sẽ tính lại tổng kiểm tra, và so sánh với tổng kiểm tra của mục dữ liệu đó trong CSDL và đưa ra thông báo lỗi.

Khi đưa ra dữ liệu trả cho người dùng, *TFE* nhận được dữ liệu từ DBMS không tin cậy, nó sẽ kiểm tra tem gắn với mục dữ liệu xem có chính xác

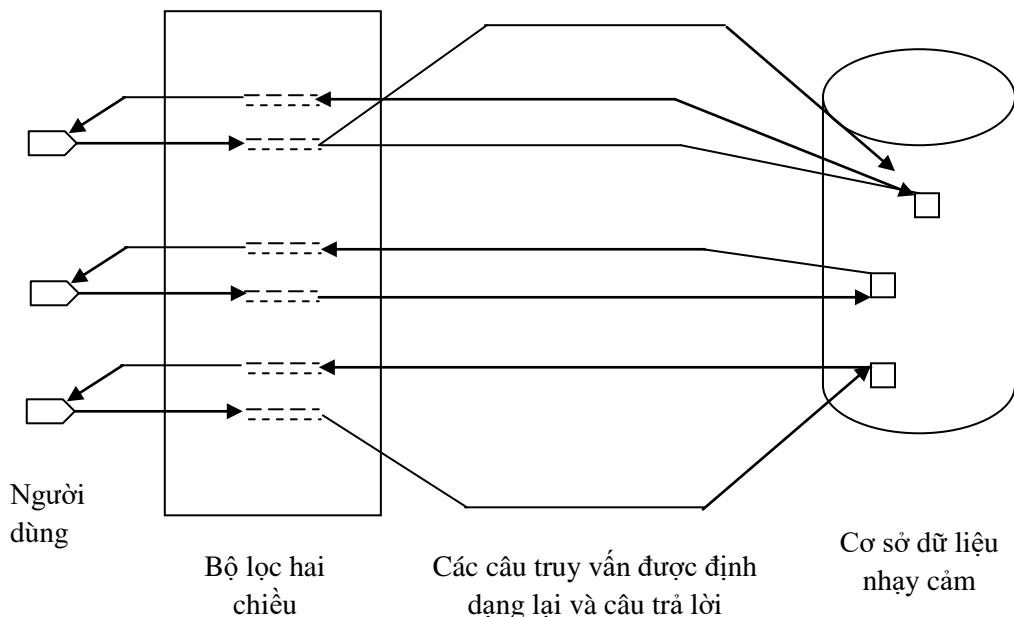
không: bằng cách giải mã tem gắn với dữ liệu, rồi so sánh với tem vừa tính được từ dữ liệu và mức nhạy cảm của nó. Nếu không khớp chứng tỏ dữ liệu đã bị sửa đổi.

TFE cũng tạo ra các bản ghi kiểm toán có khuôn dạng giống như các bản ghi kiểm toán do hệ điều hành sinh ra.

Kiến trúc này được trình bày trong hình sau đây:



Hình 3.5 Kiến trúc Integrity Lock



Hình 3.6 Bộ lọc

Khái niệm về bộ lọc giao hoán do Denning đề xuất như một sự đơn giản hoá giao diện tin cậy cho hệ quản trị cơ sở dữ liệu. Một cách cơ bản, bộ lọc sẽ che câu hỏi của người sử dụng, định dạng lại nó nếu cần để chỉ có dữ liệu có mức nhạy cảm thích hợp là được trả về cho người sử dụng.

Lọc giao hoán là quá trình tương tác với cả người sử dụng và hệ quản trị cơ sở dữ liệu. Tuy nhiên, không giống với quá trình giao tiếp ngoại vi tin cậy của khoá toàn vẹn DBMS của Graubert, bộ lọc cố gắng tận dụng hiệu quả của hầu hết DBMS. Nó sẽ định dạng lại câu truy vấn để hệ quản trị cơ sở dữ liệu có thể đảm nhiệm nhiều việc tối mức có thể, che chắn những bản ghi không cho phép truy nhập. Sau đó, bộ lọc sẽ đưa ra bức màn chắn thứ hai để chỉ chọn những dữ liệu nào mà người sử dụng được truy nhập. Có thể dùng bộ lọc để bảo đảm an toàn ở mức các bản ghi, ở mức phần tử hay thuộc tính.

Khi được dùng ở mức bản ghi, các bộ lọc yêu cầu dữ liệu bổ sung thêm thông tin về tổng kiểm tra mật mã; sau đó nó sẽ xác minh tính chính xác và khả năng truy nhập của dữ liệu được chuyển cho người sử dụng.

Ở mức thuộc tính, các bộ lọc sẽ kiểm tra xem liệu tất cả các thuộc tính trong câu truy vấn của người sử dụng có cho người sử dụng truy nhập hay không, nếu có thì chuyển câu truy vấn đến DBMS. Khi trả lời, nó sẽ xoá tất cả các trường mà người sử dụng không có quyền truy nhập.

Ở mức phần tử, hệ thống sẽ yêu cầu dữ liệu đang cần thiết cộng với thông tin về tổng kiểm tra mật mã. Khi trả lời, nó sẽ kiểm tra mức phân loại nhạy cảm của mỗi phần tử của mỗi bản ghi đã truy nhập có trái với mức của người sử dụng ko?

Một ví dụ về một câu truy vấn đơn giản:

```
retrieve NAME where ((OCCUP = PHYSICIST)^^(CITY=WASHDC))
```

Bộ lọc giao hoán sửa đổi câu truy vấn ban đầu theo cách có thể tin cậy được để thông tin nhạy cảm không bao giờ bị rút ra khỏi cơ sở dữ liệu. Câu truy vấn mẫu sẽ trở thành:

```
retrieve NAME where ((OCCUP = PHYSICIST)^^(CITY=WASHDC))  
from all record R where
```

```
(NAME_SECURITY_LEVEL(R) <= USER_SECRECY-LEVEL)^  
(OCCUP_SECRECY-LEVEL(R) <= USER_SECRECY-LEVEL)^  
(CITY_SECRECY-LEVEL(R) <= USER_SECRECY-LEVEL))
```

Bộ lọc làm việc bằng cách hạn chế câu truy vấn đến hệ quản trị cơ sở dữ liệu và sau đó hạn chế kết quả trước khi trả nó cho người sử dụng. Trong trường hợp này bộ lọc sẽ yêu cầu các giá trị NAME, NAME_SECRECY-LEVEL, OCCUP, OCCUP_SECRECY-LEVEL, CITY và CITY_SECRECY-LEVEL và sau đó sẽ lọc và trả về cho người sử dụng chỉ những bản ghi nào có mức an toàn chấp nhận được đối với người sử dụng. Mặc dù câu truy vấn mẫu này trở nên phức tạp do có thêm các thuật ngữ, song những thuật ngữ này đều đưa vào phần cuối của bộ lọc nên chúng vô hình với người sử dụng.

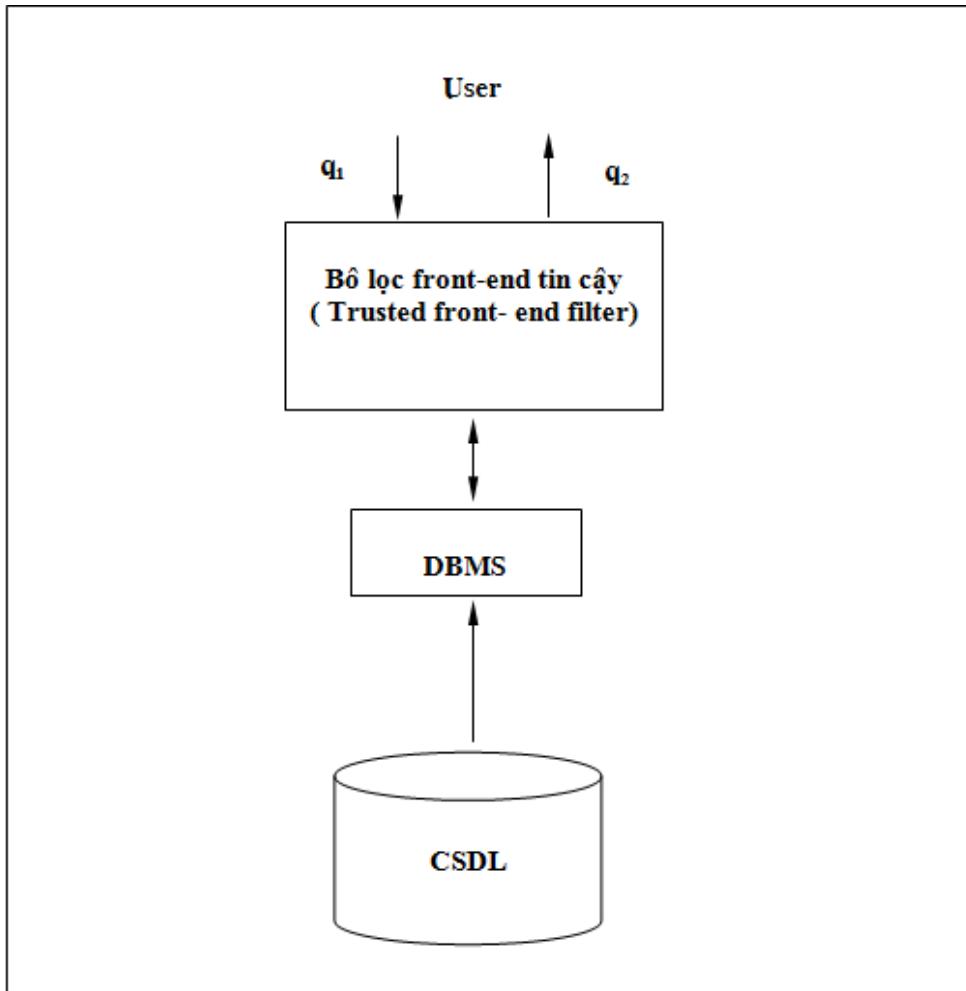
Tuy nhiên, dù tuân theo cơ chế dựa vào tem (*stamp-based mechanism*) một cách chính xác, thì cũng chưa đủ đảm bảo an toàn. Trong thực tế, nó chỉ đảm bảo cho các trường hợp sau không xảy ra:

- ✓ Truy nhập trực tiếp vào dữ liệu không được phép
- ✓ Chuyển các thông tin không được phép vào các lớp phân loại không chính xác (thông qua con ngựa thành Tōroa).

Với kiểu kiến trúc này, để tránh được các đe doạ trên, các phép chọn (*selections*), phép chiếu (*projections*), xử lý câu truy vấn phụ (*subquery handling*), tối ưu câu truy vấn (*query optimization*) và các phép thống kê (*statistical operations*) phải được cài vào trong TFE hoặc UFE, không được cài vào DBMS, DBMS chỉ có trách nhiệm đối với các phép toán lưu dữ liệu và lấy dữ liệu.

Giải pháp để loại bỏ khả năng suy diễn: đã được đề xuất năm 1985, trong đó, một bộ lọc thay thế (*commutative filter*) được chèn vào giữa DBMS và người dùng, đảm bảo loại trừ được các đe doạ suy diễn, DBMS tránh được con ngựa thành Tōroa. Giải pháp này xuất phát từ giải pháp *Maximal Authorized View* (1977).

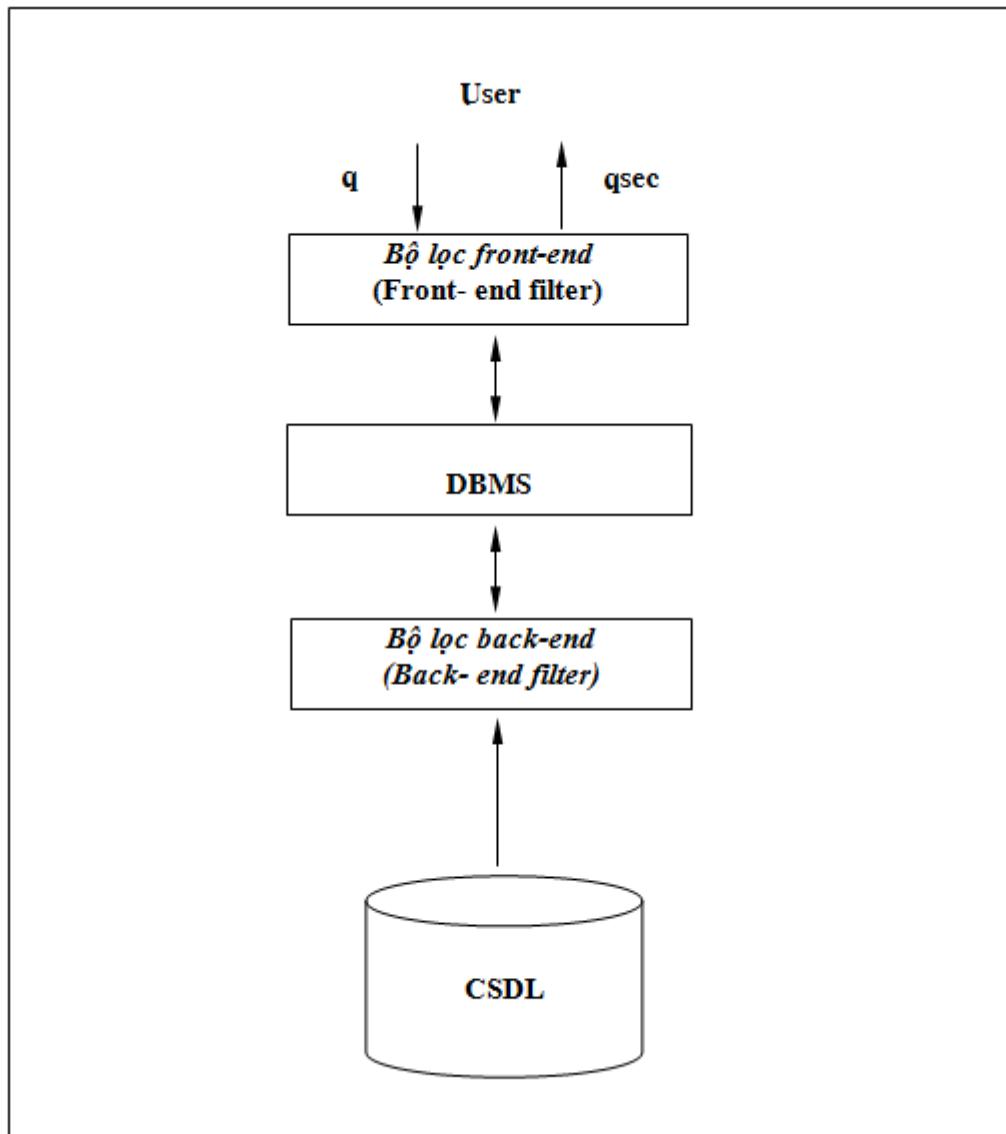
Theo giải pháp Maximal Authorized View, mỗi câu truy vấn q được ước lượng dựa vào một khung nhìn của CSDL, CSDL này chỉ bao gồm dữ liệu mà người dùng đã biết (được gọi là khung nhìn được phép tối đa, nó là một tập hợp con của dữ liệu được lưu giữ trong CSDL), nó đưa ra nguồn dữ liệu cho một câu truy vấn q_{sec} , tránh suy diễn trên dữ liệu không được phép.



Hình 3.7 Giải pháp bộ lọc thay thế.

Bộ lọc back-end (còn được gọi là bộ lọc quản lý dữ liệu) có trách nhiệm trong việc định nghĩa khung nhìn được phép tối đa, bằng cách phát hiện tất cả các bản ghi/các thuộc tính không được phép, thay thế các yếu tố không được phép bằng giá trị 0.

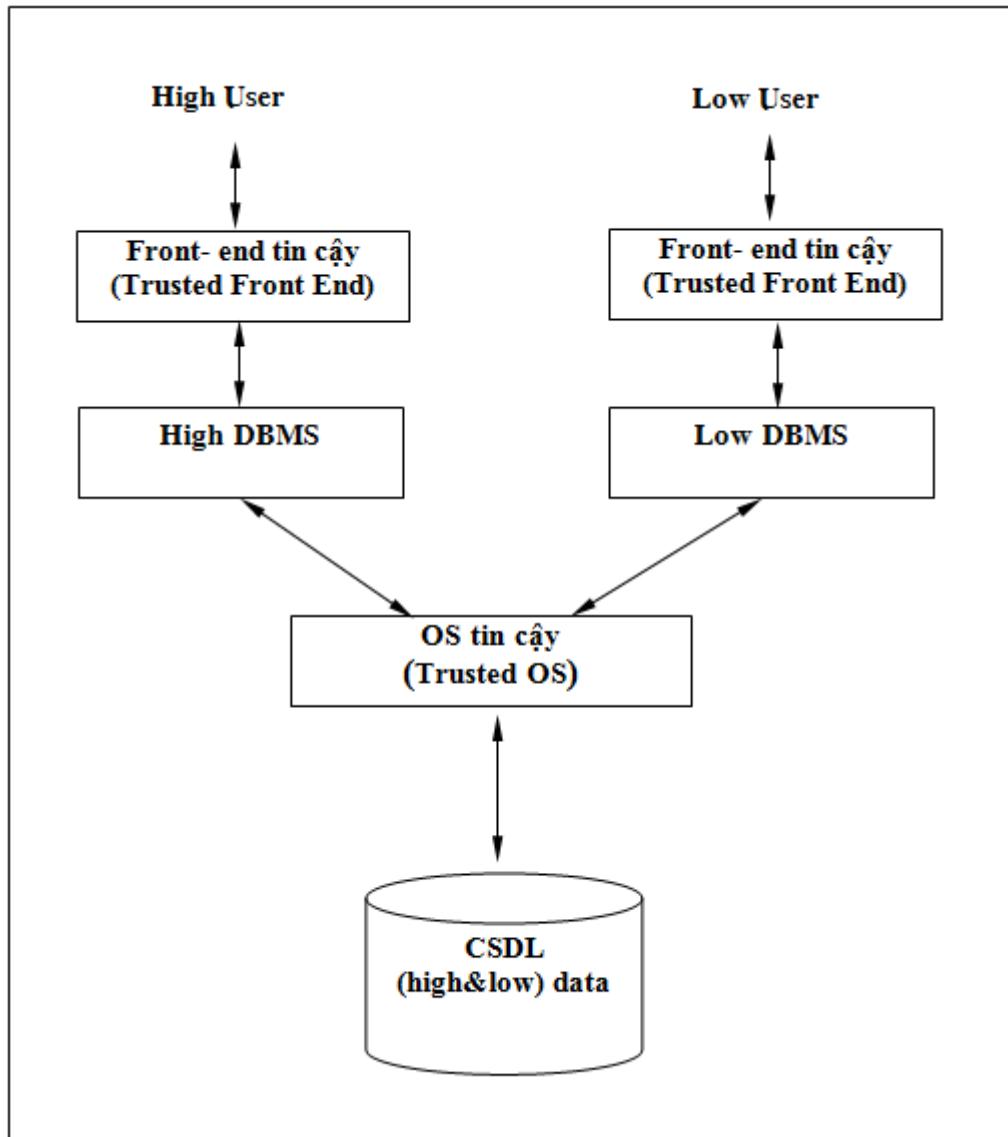
Bộ lọc front-end của kiến trúc được trình bày trong hình 3.7 làm việc theo cách như vậy, câu truy vấn q_2 (được trả lại cho người dùng) tương đương với câu truy vấn q_{sec} của kiến trúc trong hình 3.8, bằng cách lọc kết quả của câu truy vấn q_1 .



Hình 3.8 Giải pháp khung nhìn cho phép tối đa

❖ **Kiến trúc Kernelized**

Kiến trúc này được trình bày trong hình 3.9.



Hình 3.9 Kiến trúc Kernelized

Ở đây có sử dụng một OS tin cậy, nó có trách nhiệm đối với các truy nhập vật lý vào dữ liệu (trong CSDL) và có trách nhiệm tuân theo bảo vệ bắt buộc. High User (người dùng làm việc ở mức cao) tương tác với một High DBMS, thông qua một TFE, Low User (người dùng làm việc ở mức thấp) tương tác với một Low DBMS cũng thông qua một TFE. Sau đó, các yêu cầu của họ được chuyển cho OS, và OS lấy lại dữ liệu hợp lệ từ CSDL.

Theo giải pháp này, các đối tượng (có các nhãn an toàn giống nhau) của CSDL được lưu giữ trong các đối tượng của OS tin cậy (đóng vai trò như là

các kho chứa đối tượng của CSDL). Vì vậy, OS tin cậy tiến hành kiểm soát an toàn trên các đối tượng lưu giữ này, cần có các quá trình phân tách và khôi phục quan hệ nhiều mức.

Quá trình phân tách: được thực hiện khi chuyển đổi một quan hệ đa mức thành một số quan hệ đơn mức, (chỉ chứa dữ liệu ở một mức an toàn nào đó), chúng được lưu giữ trong các đối tượng của hệ điều hành. Quá trình này được thực hiện khi OS tiến hành lưu giữ các đối tượng CSDL vào các đối tượng của nó.

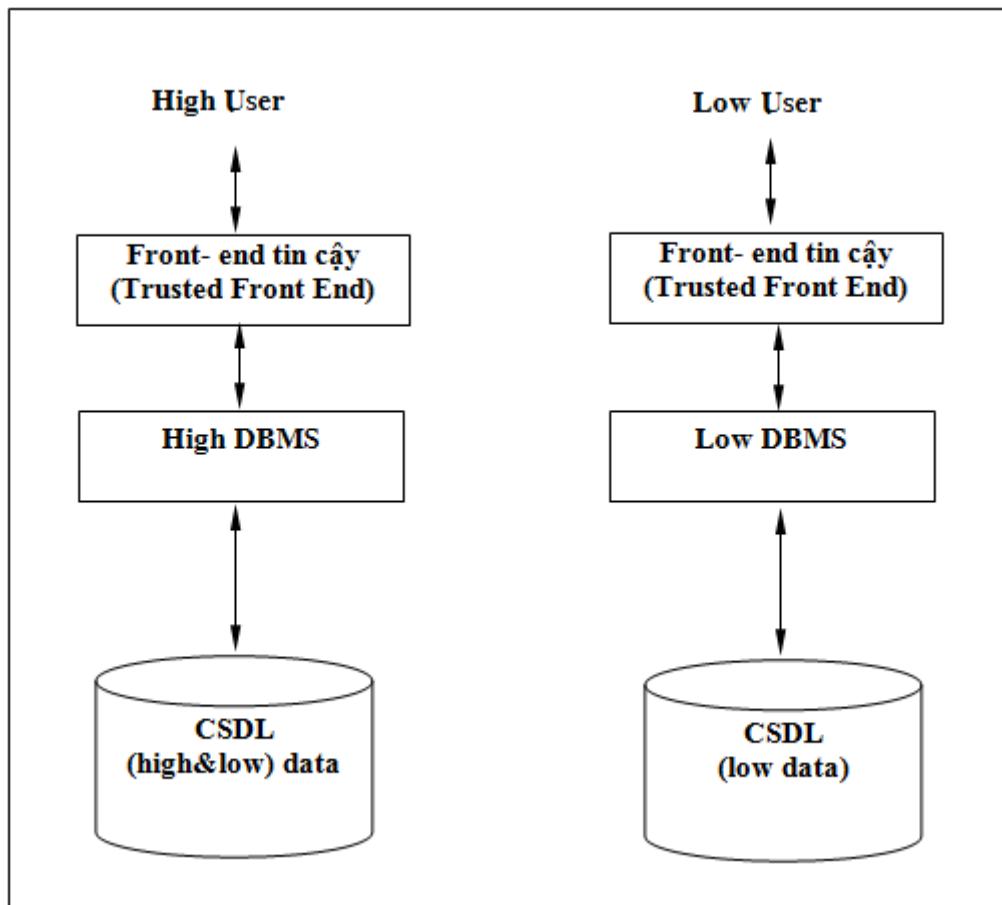
Quá trình khôi phục: được thực hiện khi OS cần lấy lại dữ liệu được lưu giữ trên các đối tượng của nó để trả về cho người dùng. Để từ các quan hệ đơn mức, sinh ra một khung nhìn đa mức chỉ chứa các dữ liệu mà người dùng yêu cầu,

Các thuật toán phân tách và khôi phục phải được định nghĩa chính xác, nhằm đảm bảo tính đúng đắn và hiệu quả của hệ thống. Các bản ghi kiểm toán được OS tin cậy sinh ra cho các phép toán liên quan đến truy nhập vào các đối tượng của OS. Kiến trúc này được sử dụng trong mẫu thử nghiên cứu Sea View và DBMS Oracle thương mại.

❖ **Kiến trúc Replicated (lặp)**

Kiến trúc này được trình bày trong hình 3.10. Theo giải pháp này, *dữ liệu mức thấp được lặp trong CSDL*. Theo cách này:

- ✓ Người dùng mức thấp chỉ được phép truy nhập vào CSDL độ ưu tiên thấp, không có khả năng sửa đổi dữ liệu mức cao.
- ✓ Người dùng mức cao có thể xem và sửa đổi cả dữ liệu mức thấp và mức cao.
- ✓ Để tuân theo giải pháp này cần có các thuật toán đồng bộ an toàn để đảm bảo tính tương thích lặp và chi phí (do lặp) tăng dần theo kích cỡ của lưới an toàn.
- ✓ Không một DBMS thương mại nào sử dụng kiến trúc này vì nó rất đắt, do phải lặp dữ liệu. Nó chỉ được sử dụng trong mẫu thử nghiên cứu NRL.



Hình 3.10 Kiến trúc Replicated

Nhận xét về các kiến trúc an toàn

Khác nhau về môi trường ứng dụng: các kiến trúc an toàn được trình bày ở trên thích hợp cho các mục đích khác nhau, tuỳ thuộc vào các đặc điểm và các yêu cầu của miền ứng dụng đích. Ví dụ:

- ✓ *Kiến trúc Kernelized* phù hợp với các môi trường có yêu cầu bảng đơn mức, bởi vì nó kinh tế nhất và dễ thực hiện nhất.
- ✓ *Kiến trúc Integrity Lock*: phù hợp với những môi trường đặc trưng bởi một DBMS yêu cầu tính mềm dẻ của nhãn và một mức tích hợp cao giữa DBMS và OS cơ sở.

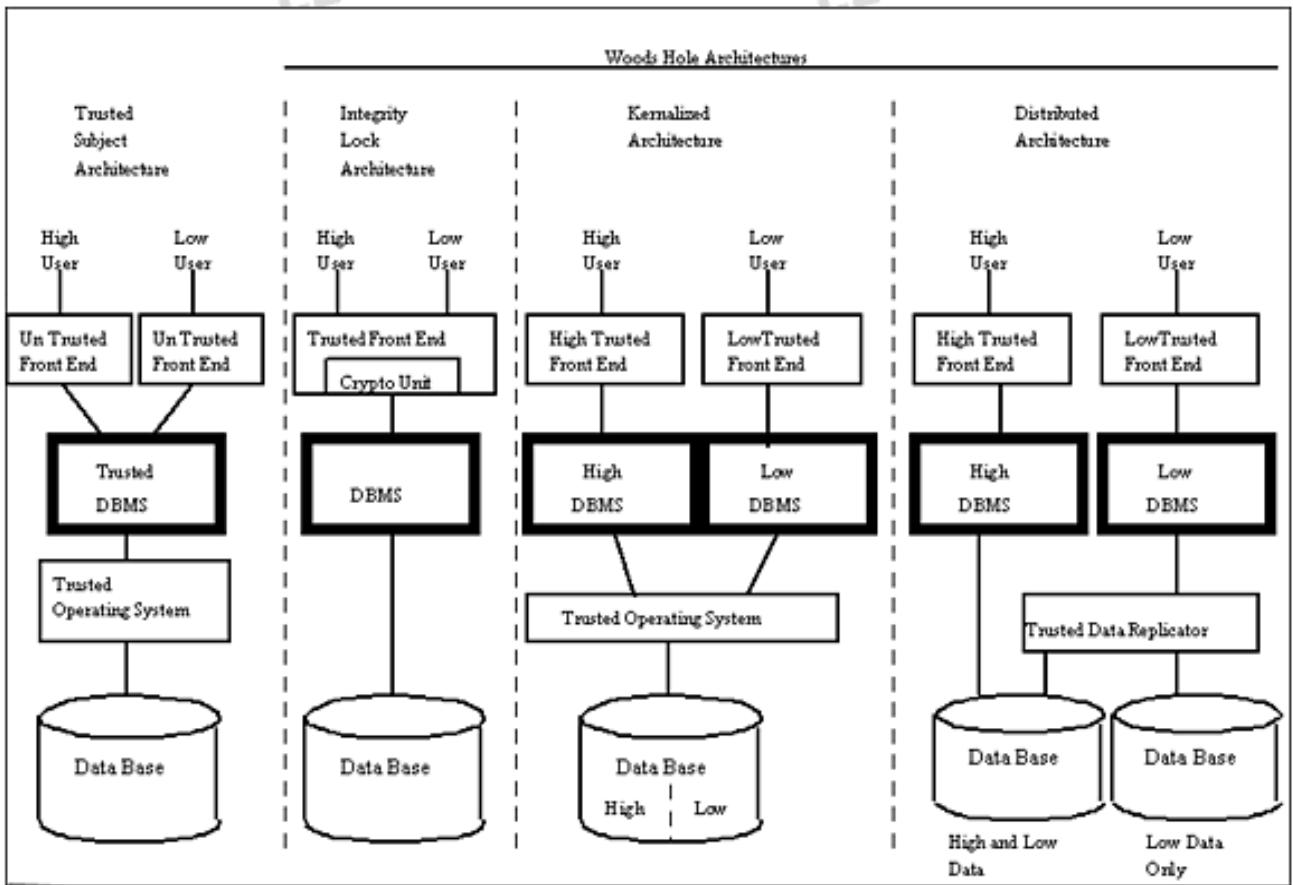
- ✓ *Kiến trúc chủ thẻ tin cậy*: thích hợp với các miền ứng dụng cần đảm bảo một đường dẫn tin cậy từ các ứng dụng đến DBMS.

Khác nhau khi đánh giá mức tin cậy của các kiến trúc: độ phức tạp trong vấn đề đánh giá phụ thuộc vào kiến trúc. Ví dụ:

- ✓ Kiến trúc Integrity Lock: được phê chuẩn một cách dễ dàng nhất
- ✓ Kiến trúc chủ thẻ tin cậy thì phức tạp hơn.

Thực vậy, đối với kiến trúc Integrity Lock sẽ dễ dàng đánh giá hơn chỉ với một bộ lọc kích cỡ nhỏ, trong khi kiến trúc Chủ thẻ tin cậy phải đánh giá một DBMS tin cậy, bao gồm việc đánh giá các dịch vụ thông thường do một OS tin cậy cung cấp.

- ✓ Kiến trúc Kernelized nằm ở vị trí trung gian, nhưng nếu phải bổ sung thêm phần mềm tin cậy nhằm đảm bảo hoạt động an toàn trong một môi trường đa mức, thì việc đánh giá trở nên khó khăn hơn.
- Khác nhau về mức độ phụ thuộc giữa DBMS và OS cơ sở tin cậy:
 - ✓ Các kiến trúc Integrity Lock và Kernelized dựa vào các dịch vụ an toàn do OS cơ sở tin cậy cung cấp
 - ✓ Kiến trúc chủ thẻ tin cậy đưa ra một mức phụ thuộc và tích hợp thấp hơn. - *Khác nhau về độ chi tiết của đối tượng được gán nhãn*: đối tượng nhỏ nhất của CSDL có thể được gán nhãn của các kiến trúc khác nhau thì khác nhau.
 - ✓ Kiến trúc Integrity Lock và kiến trúc Chủ thẻ tin cậy cung cấp khả năng gán nhãn mức hàng. Hai kiến trúc này có thể được mở rộng, nhằm hỗ trợ cho việc gán nhãn tại mức trường của một hàng,
 - ✓ Kiến trúc Kernelized: việc gán nhãn do OS tin cậy cung cấp, và gán nhãn trên các đối tượng chứa của nó (chứa các đối tượng CSDL), vì vậy giảm tổng chi phí lưu trữ. Tuy nhiên, cơ chế gán nhãn này sẽ không thích hợp nếu cần phải quản lý nhiều các bảng đa mức. Kiến trúc Kernelized không hỗ trợ gán nhãn đến mức trường của một hàng như 2 kiến trúc trên.



Hình 3.11. Tổng quan các kiến trúc DBMS nhiều mức như sau

3.2 GIỚI THIỆU MỘT SỐ HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

3.2.1 Sơ lược kiến trúc một số hệ quản trị

Một hệ quản trị cơ sở dữ liệu là một bộ phần mềm cung cấp cho người dùng các chức năng quản lý, phân tích, và tra cứu dữ liệu. Hầu hết các hệ quản trị cơ sở dữ liệu ngày nay được thiết kế để làm việc với các dữ liệu quan hệ. Những hệ quản trị này được gọi là hệ quản trị cơ sở dữ liệu quan hệ (RDBMS - Relation Database Management Systems).

Một số hệ quản trị cơ sở dữ liệu quan hệ thường được biết đến như: Oracle, Microsoft SQL Server, MySQL, PostgreSQL, DB2, SYBASE, Infomix... Trong đó các hệ quản trị cơ sở dữ liệu phát triển mạnh nhất là Oracle, Microsoft SQL Server, MySQL.

Oracle

Oracle là hệ quản trị cơ sở dữ liệu được phát triển bởi “Oracle Corporation” vào cuối những năm 1970. Năm giữ 52% thị phần quản trị cơ sở dữ liệu vào năm 2009, nó còn là một trong những cơ sở dữ liệu phổ biến nhất trên các máy chủ. Hệ quản trị cơ sở dữ liệu Oracle được biết đến rộng rãi với sự linh hoạt, khả chuyển, nó có thể chạy trên hầu hết các hệ điều hành. Oracle mang đến các giải pháp lưu trữ, duy trì các dữ liệu kinh doanh quan trọng như tài nguyên con người, thanh toán và báo cáo tài chính trên khắp thế giới.

Các phiên bản phổ biến của Oracle trong những năm gần đây: Oracle 10g, Oracle 11g R2.

MySQL

MySQL là phần mềm cơ sở dữ liệu mã nguồn mở phổ biến nhất thế giới, với hơn 100 triệu bản phần mềm được tải về hoặc phân phối trong suốt lịch sử của nó. Với tốc độ cao, độ tin cậy, và dễ sử dụng, MySQL đã trở thành sự lựa chọn ưa thích cho các ứng dụng Web, Web 2.0, các công ty viễn thông bởi vì nó giúp loại bỏ những vấn đề chính liên quan đến thời gian tải, bảo trì và quản trị cho các ứng dụng trực tuyến, hiện đại.

Nhiều tổ chức lớn và phát triển trên thế giới sử dụng MySQL để tiết kiệm thời gian và tiền bạc cho các trang web lớn của họ. Các hệ thống kinh doanh quan trọng, và đóng gói phần mềm - bao gồm các doanh nghiệp dẫn đầu như Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube, Wikipedia và Booking.com cũng sử dụng MySQL. MySQL là một phần quan trọng của LAMP (Linux, Apache, MySQL, PHP/ Perl/ Python). Ngày càng có nhiều công ty sử dụng LAMP để thay thế cho các phần mềm độc quyền bởi vì nó có chi phí rẻ hơn và tự do lựa chọn nền tảng.

Dự án MySQL được phát triển và công bố mã nguồn theo các điều khoản của GNU GPL, cũng như một loạt các thỏa thuận độc quyền. Ban đầu MySQL thuộc sở hữu và được tài trợ bởi một công ty phi lợi nhuận duy nhất, công ty MySQL AB Thụy Điển, ngày nay MySQL thuộc sở hữu của tập đoàn Oracle.

Các phiên bản phổ biến của MySQL trong những năm gần đây: MySQL 5.5, MySQL 5.6.

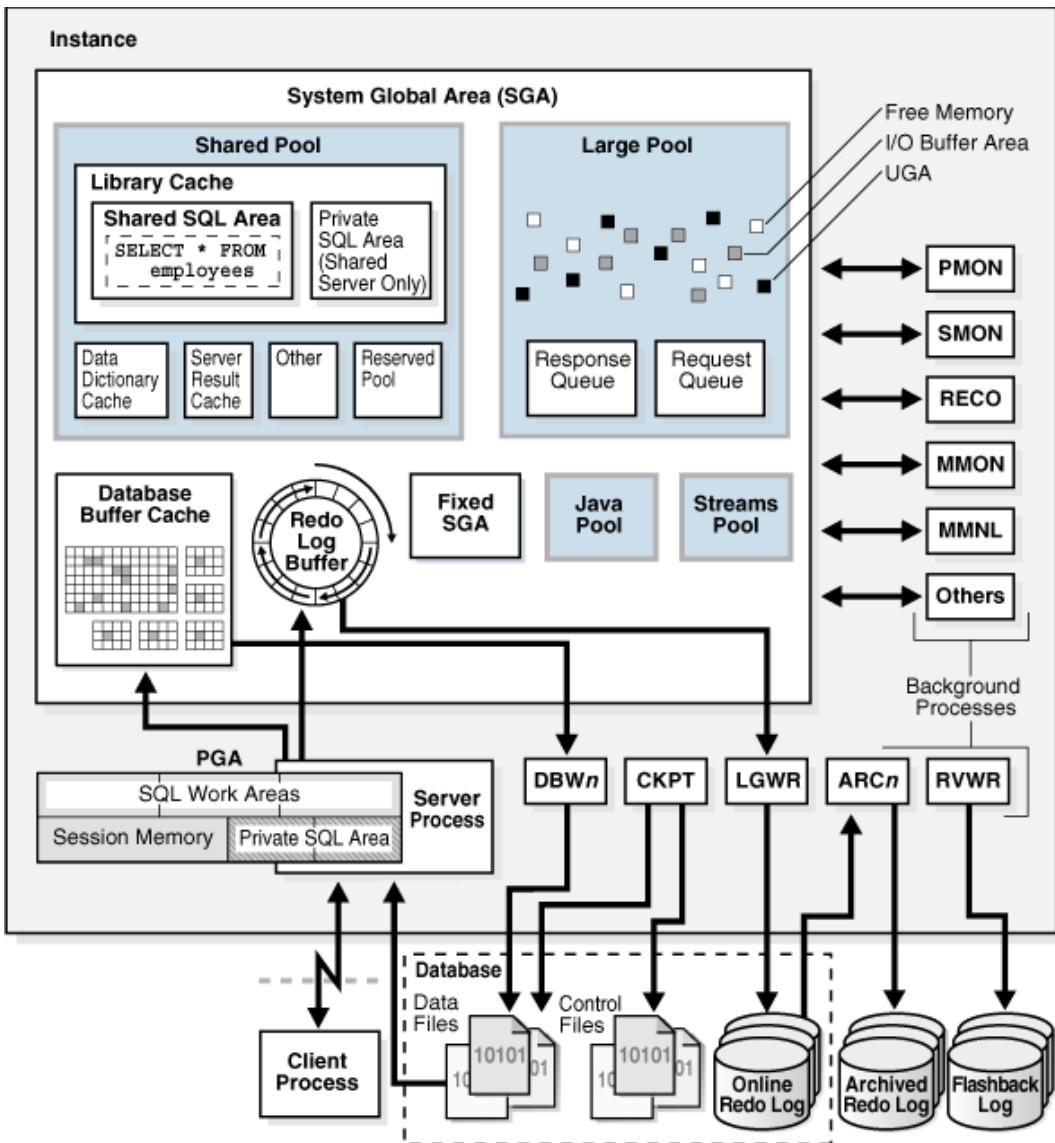
Microsoft SQL

Microsoft SQL hay thường được gọi đơn giản là SQL Server. SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ được phát triển bởi Microsoft để cung cấp một nền tảng truy xuất dữ liệu nhanh, an toàn và có khả năng mở rộng. Ngôn ngữ truy vấn dữ liệu chính của SQL Server là T-SQL và ANSI SQL. Khả năng mở rộng của SQL Server là thành phần hấp dẫn nhất, được phát triển để chạy trong nhiều môi trường khác nhau. Nó có thể đáp ứng nhu cầu của bất kỳ môi trường Windows nào, từ cơ sở dữ liệu chạy trên máy cá nhân chỉ có một user, cho tới cơ sở dữ liệu được lưu trữ trên một cụm máy chủ phục vụ hàng nghìn users.

Các phiên bản phổ biến của SQL Server trong những năm gần đây: SQL Server 2005, SQL Server 2008, SQL Server 2008 R2, SQL Server 2012.

3.2.1.1 Kiến trúc của Oracle

Kiến trúc của Oracle 11g R2 được miêu tả đầy đủ trong tài liệu “*Oracle® Database Concepts 11g Release 2 (11.2) E25789-01*” của hãng Oracle.



Hình 3.12 Oracle Instance và Database

Instance và Database

Một “instance” (thể hiện) là một thuật ngữ để cập đến các tiến trình nền và cấu trúc bộ nhớ được sử dụng trong suốt quá trình tương tác với cơ sở dữ liệu. Để tạo một instance, người dùng phải kết nối tới cơ sở dữ liệu và thiết lập phiên kết nối. Một phiên (session) được bắt đầu khi người dùng kết nối tới cơ sở dữ liệu và kết thúc khi người dùng đăng xuất (logout) khỏi cơ sở dữ liệu. Tập các tiến trình nền trong một instance gồm có tiến trình giám sát hiệu năng (PMON - Performance Monitor), giám sát hệ thống (SMON - System

Monitor), khôi phục (RECO - Recovered), ghi nhật ký (LGWR - Log Writer), và kiểm tra thời điểm (CKPT - Checkpoint).

Có hai kiến trúc bộ nhớ quan trọng trong Oracle instance: SGA - the System Global Area, và PGA - the Program Global Area. Những nguồn tài nguyên này đóng vai trò quan trọng với độ tin cậy và hiệu năng của Oracle.

Phần cơ sở dữ liệu của một máy chủ Oracle quản lý những file cơ sở dữ liệu tạo môi trường cần thiết để chạy cơ sở dữ liệu Oracle. Đó là tập các file giúp cấu hình instance, giúp nó có thể thực thi và xử lý các câu truy vấn SQL, đảm bảo việc cảnh báo và phục hồi khi có lỗi từ phần cứng và phần mềm. Ví dụ các file trong khu vực này là control, redo log, và data.

Kiến trúc vật lý

Kiến trúc vật lý phụ thuộc vào hệ điều hành mà Oracle được cài đặt, và có một số file bắt buộc với mọi kiểu cài đặt Oracle. Các file chứa bên trong kiến trúc vật lý tương tác với hệ điều hành và trong suốt với người dùng. Những file phổ biến đó là datafile, control file, và redo log file.

- **Datafile:** File này chứa dữ liệu thật sự cho cơ sở dữ liệu và giữ thông tin cho tất cả các kiến trúc logic (tables, records, ...) trong cơ sở dữ liệu.

- **Control file:** File này chứa vị trí và chứng chỉ thông tin quan trọng của các file khác, đóng vai trò quan trọng với chức năng của cơ sở dữ liệu. Do nó chứa thông tin vị trí cho các file quan trọng khác, nên nếu file này bị hỏng, cơ sở dữ liệu sẽ không chạy được. Ví dụ các thông tin được giữ trong control file là: trạng thái đọc/ghi của các file, ID duy nhất của cơ sở dữ liệu, thời điểm bị lỗi nếu xảy ra lỗi, tên và vị trí của datafile. Việc lưu bản sao của control file nên được thực hiện nhằm dự phòng khi có lỗi xảy ra.

- **Redo log file:** File này chứa tất cả các thay đổi tác động tới dữ liệu trong cơ sở dữ liệu. Nó tương tự như một nút bấm “undo” cho cơ sở dữ liệu. File redo log có thể được sử dụng để phục hồi dữ liệu trong trường hợp dữ liệu bị mất. Tất cả kiểu cài đặt của Oracle đều bao gồm

ít nhất hai bản sao của redo log, và việc lưu bản sao để phòng lỗi xảy ra là một thông lệ tốt.

Kiến trúc bộ nhớ

Như chúng ta đã biết, bộ nhớ trong và bộ đệm là các thiết bị lưu trữ có tốc độ truy xuất nhanh nhất trong hệ thống. Vì vậy một tiến trình chạy trực tiếp từ bộ nhớ sẽ đảm bảo hệ thống được hiệu quả và tin cậy.

Trong Oracle, thực tế mọi thứ xảy ra bên trong bộ nhớ chính, ngay cả khi người dùng kết nối đến bộ nhớ qua tiến trình của máy chủ. Đây là một trong những lý do khiến hệ quản trị cơ sở dữ liệu quan hệ Oracle có khả năng phục vụ tin cậy cho hàng trăm tới hàng nghìn người dùng trong cùng một thời điểm.

Một ví dụ về cách Oracle tối ưu bộ nhớ để tăng tốc độ xử lý là “query caching” (bộ đệm truy vấn). **Caching** là một tiến trình lưu lại những yêu cầu dữ liệu trùng nhau sang một vùng khác của hệ thống, với hy vọng tiết kiệm tài nguyên và tăng tốc cho những yêu cầu phía sau cho cùng dữ liệu đó. Trong một cơ sở dữ liệu Oracle, các truy vấn được đệm sang vùng buffer để tiếp tục tối ưu bộ nhớ, sau đó tăng tốc độ trả về cho các truy vấn sau này. Kiến trúc bộ nhớ của Oracle được chia thành hai phần chính: vùng toàn cục hệ thống (SGA - System Global Area) và vùng toàn cục tiến trình (PGA - Process Global Area).

System Global Area (SGA): là vùng trung tâm mà tất cả các dữ liệu chia sẻ và dữ liệu xử lý được lưu trữ. Thông tin được chứa trong SGA bao gồm thông tin chia sẻ bởi người dùng và các tiến trình của cơ sở dữ liệu. SGA cũng lưu giữ việc kiểm soát dữ liệu cho một “single instance” trong Oracle. Oracle 11g sử dụng SGA động, nghĩa là dữ liệu có thể bị thay đổi và SGA cũng tạo ra một instance mới cho Oracle. SGA chứa một số các kiến trúc bộ nhớ bắt buộc và tùy chọn. Bảng dưới đây miêu tả các kiến trúc bộ nhớ bắt buộc và tùy chọn trong SGA.

Kiến trúc	Miêu tả	Thông tin và ví dụ
-----------	---------	--------------------

Database buffer cache	Được sử dụng để đệm thông tin đọc từ file dữ liệu cũng như các truy vấn SQL và PL/SQL được sử dụng gần nhất.	Một người dùng thực thi truy vấn trên một máy trạm, và máy trạm kết nối với tiến trình trên máy chủ. Đầu tiên, tiến trình máy chủ kiểm tra bộ đệm xem file đã tồn tại hay chưa. Nếu đã tồn tại, nó sẽ trả kết quả về cho người dùng; nếu nó không có trong bộ đệm, tiến trình máy chủ lấy dữ liệu trong datafile và tải vào bộ đệm buffer để sử dụng trong các lần sau.
Shared pool	Lưu trữ hầu hết các câu truy vấn SQL được thực hiện gần nhất và các định nghĩa dữ liệu có chứa “Library cache” và “Data dictionary cache”.	Khi ta thực thi câu truy vấn SELECT, câu truy vấn được thực thi một lần, sau đó được lưu trong Shared pool; các quyền của người dùng được kiểm tra để chắc chắn người dùng có thể xem các thông tin yêu cầu, sau đó loại bỏ thông tin được yêu cầu từ datafile, tải nó vào bộ đệm buffer cho người dùng hiển thị.
Library cache	Được dùng để đệm các thông tin siêu dữ liệu (metadata).	Một câu lệnh SQL được lưu trữ ở đây đã được phân tích cú pháp trong khi cú pháp đã được kiểm tra bởi cơ sở dữ liệu; sau khi được xác nhận hợp lệ, nó tìm kiếm trong Share pool bản đã được đệm của câu lệnh.
Data dictionary cache	Đệm các thông tin được sử dụng gần nhất bởi Data dictionary	Thông tin tài khoản người dùng, tên datafile, v.v...

Database buffer cache	Lưu các khối của bảng và dữ liệu đã nhận được từ trước	Một truy vấn được gửi tới tiến trình máy chủ; tiến trình máy chủ đầu tiên tìm kiếm trong Database buffer để tìm thông tin cần thiết để truy nhập ổ cứng, do đó làm tăng hiệu năng
Redo log buffer	Lưu trữ tất cả các thay đổi tác động tới cơ sở dữ liệu	Khi redo buffer bị đầy, những phần thừa được chuyển vào redo file.

Bảng 3.2 Các kiến trúc bộ nhớ bắt buộc có trong SGA

Kiến trúc	Miêu tả	Thông tin và ví dụ
Large pool	Lưu trữ những công việc lớn tránh làm đầy shared pool	RMAN backups
Java pool	Lưu trữ và đệm các câu lệnh Java	Chỉ cần thiết khi Java được cài đặt
Streams pool	Cho hàng đợi cao cấp	Chỉ được dùng trong Oracle 10g và 11g

Bảng 3.2 Các kiến trúc bộ nhớ tùy chọn trong SGA

Các tiến trình nền

Một tiến trình là một tập các chỉ dẫn được thực thi bởi hệ điều hành để hoàn thành một nhiệm vụ nào đó. Trong Oracle, để hoàn thành một instance, có hai việc cần thực hiện. Đầu tiên, người dùng cần chạy một công cụ ứng dụng như SQL, để yêu cầu kết nối đến máy chủ Oracle. Đây được gọi là tiến trình người dùng. Tiếp theo, máy chủ phải xử lý yêu cầu của người dùng và chạy một tiến trình để tạo Oracle instance và hoàn thành kết nối, đây được gọi

là tiến trình máy chủ. Khi một Oracle instance được tạo ra, các tiến trình nền chạy để duy trì các mối liên lạc và các nguồn tài nguyên giữa các thành phần Oracle.

Các tiến trình máy chủ có thể được chia sẻ hoặc dùng riêng. Trong một môi trường máy chủ được dùng riêng, mỗi một phiên sở hữu một tiến trình máy chủ riêng để để thực thi các câu lệnh SQL và xử lý các yêu cầu người dùng. Trong một môi trường máy chủ chia sẻ, những người sử dụng phải chia sẻ các tiến trình máy chủ cho việc xử lý các yêu cầu và các câu lệnh SQL. Việc cấu hình phụ thuộc vào nhu cầu và quy mô của tổ chức. Các tổ chức lớn có thể chọn môi trường máy chủ chia sẻ để số các tiến trình phải chạy trên máy chủ là ít nhất, các tổ chức nhỏ hơn có thể ít quan tâm tới chi phí máy chủ.

Có vài tiến trình nền đi kèm với các Oracle instance. Một số là bắt buộc, số khác là các tùy chọn. Bảng sau miêu tả sự khác nhau giữa các tiến trình nền trong một Oracle instance.

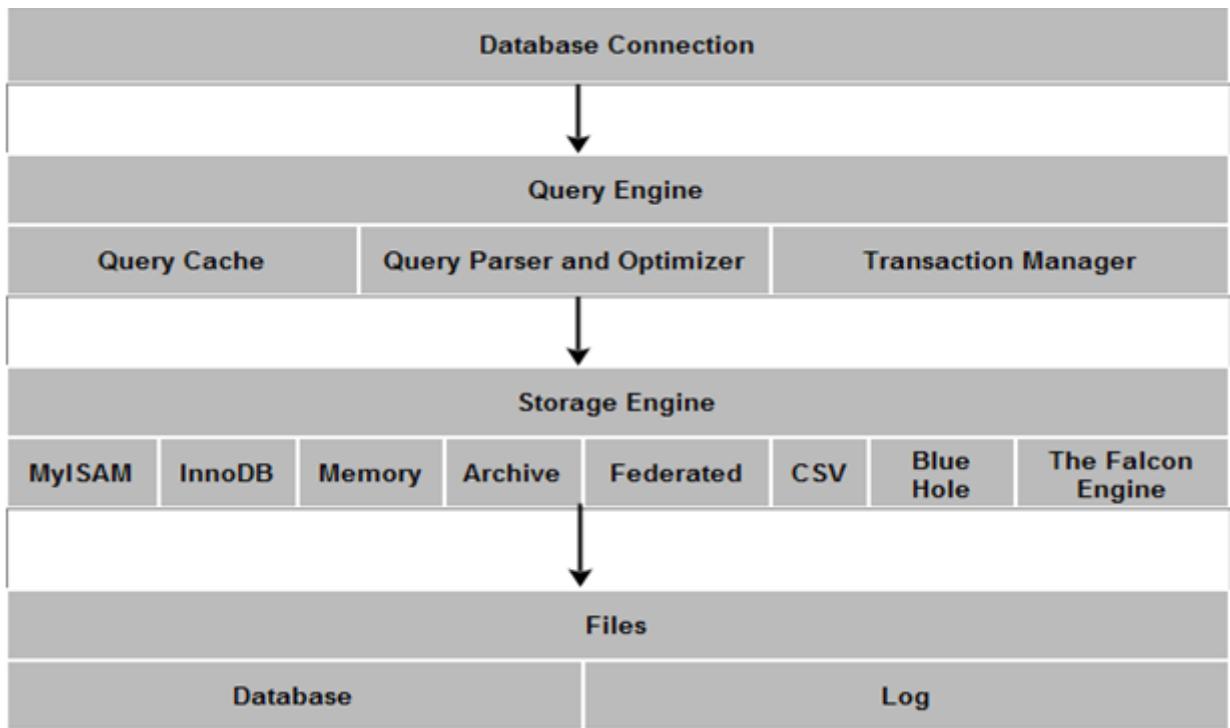
Tên tiến trình	Các công việc của tiến trình
PMON (process monitor)	Dọn dẹp sau khi các tiến trình khác hoàn thành hoặc bị lỗi.
SMON (system monitor)	Phục hồi cơ sở dữ liệu khi bị hỏng bằng cách sử dụng relog cũng như các file cơ sở dữ liệu.
DBWN (database writer)	Ghi các thay đổi từ vùng đệm cơ sở dữ liệu xuống file cơ sở dữ liệu.
LGWR (log writer)	Ghi lại những sự sửa đổi từ vùng đệm redo log xuống file redo log.
CKPT (checkpoint)	Ghi vào control file những thời điểm được thiết lập khi bắt đầu quá

trình khôi phục, nếu cần thiết.

Bảng 3.3 Một số tiến trình nền

3.2.1.2 Kiến trúc của MySQL

MySQL được phát triển để sử dụng cho đa nền tảng. MySQL có thể được tìm thấy trên Solaris, Linux, hoặc Windows. Các thành phần kiến trúc được định nghĩa trong phần này là những thành phần không thay đổi trên cả ba nền tảng hệ điều hành trên. Các cấu trúc chính của kiến trúc máy chủ MySQL gồm có: bộ quản lý kết nối cơ sở dữ liệu (database connection manager), engin truy vấn (query engine), bộ quản lý giao dịch (transaction manager) và engin lưu trữ (storage engine).



Hình 3.13 Kiến trúc MySQL

Bộ quản lý kết nối cơ sở dữ liệu (Database Connection Manager)

Đây là trình quản lý các kết nối tới máy chủ MySQL. Lớp quản lý kết nối rất linh hoạt, kích hoạt giả lập bất kỳ máy trạm kết nối đến máy chủ MySQL. Một ứng dụng nguồn mở được xây dựng để chạy giả lập trên mọi nền tảng và

MySQL cung cấp một số cách để cho các máy trạm kết nối qua lớp quản lý kết nối cơ sở dữ liệu. Các nhà phát triển có thể tạo các máy trạm và các giao diện lập trình ứng dụng (API) bằng hầu hết các ngôn ngữ hiện đại ngày nay như: C, C++, Perl, PHP, và các máy trạm sử dụng kết nối cơ sở dữ liệu mở ODBC, Java, và .NET đã được cung cấp qua giao tiếp của MySQL.

Mặc dù có nhiều lựa chọn để kết nối đến MySQL qua lớp quản lý kết nối, nhưng TCP/IP là loại kết nối phổ biến nhất đã được tối ưu. TCP/IP có thể được sử dụng trong hầu hết các môi trường hệ điều hành (Windows, UNIX, OS/2 Linux và Solaris), và rất thu hút người dùng lựa chọn. Có thể coi là an toàn nếu giả định rằng hầu hết các ứng dụng kết nối đến máy chủ MySQL đang sử dụng TCP/IP.

Mỗi kết nối máy chủ sở hữu luồng (thread) riêng, MySQL gán động không gian vùng đệm trong quá trình thực hiện truy vấn. Một thread thực thi chạy độc lập với các tiến trình khác và tối ưu các phần sử dụng trên CPU. Những thread này xử lý tất cả các yêu cầu từ máy trạm trong suốt quá trình kết nối, bao gồm quá trình xác thực và xử lý truy vấn. Có nhiều thread thực thi trong cùng một thời điểm. Mặc dù thread dùng riêng cung cấp sự tiện lợi cho người dùng, nhưng nó làm cho chi phí xử lý của máy chủ tăng cao, vì thế phải chắc chắn rằng các thread đã được tối ưu một cách hiệu quả.

Ngoài quản lý các kết nối thực sự đến cơ sở dữ liệu, trình quản lý kết nối còn cho phép các máy trạm kết nối cục bộ và kết nối từ xa đến cơ sở dữ liệu, sử dụng các hồ sơ (profile) đã được định nghĩa trước. Những profile này cho phép người dùng định nghĩa vai trò cụ thể trong cơ sở dữ liệu của họ, và sau đó lưu lại vai trò đó cùng với các tham số cấu hình để sử dụng cho lần sau. Các hộp thoại profile mang đến sự tiện lợi to lớn cho người dùng, loại bỏ việc phải cấu hình lại mỗi lần kết nối.

Engine truy vấn (Query Engine)

Engine truy vấn là thành phần kiến trúc giúp tối ưu trong quản lý các truy vấn và các câu lệnh SQL. Engine truy vấn được xây dựng để sử dụng tài nguyên hiệu quả nhất, vì vậy, nó có thể hỗ trợ dữ liệu OLTP và DSS như nhau. Các truy vấn trong MySQL được viết bằng các phiên bản SQL khác

nhau để cung cấp tìm kiếm mạnh hơn. Phần này sẽ miêu tả tóm tắt việc nhận, xử lý, và tối ưu truy vấn trong engine truy vấn của MySQL.

Thông thường, khi người dùng kết nối đến MySQL qua ứng dụng máy trạm của họ thì một truy vấn được khởi tạo. Khi máy chủ MySQL nhận được một yêu cầu truy vấn, nó phản hồi truy vấn này bằng ngôn ngữ thao tác dữ liệu (DML) để tìm các câu lệnh SQL liên quan và nhận chúng từ ứng dụng phía máy trạm. Sau đó, ngôn ngữ định nghĩa dữ liệu (DDL) cung cấp các truy nhập cho SQL để tương tác với cơ sở dữ liệu.

Trước khi các truy nhập được thực hiện, bộ phân tích truy vấn tạo dạng cấu trúc cây, dựa trên việc tách các câu lệnh SQL. Sau đó, bộ phân tích kiểm tra tính hợp lệ về ngữ pháp và ngữ nghĩa của cây. Nếu câu lệnh không hợp lệ, nó trả lại một thông báo lỗi cho người dùng. Nếu câu lệnh hợp lệ, trình quản lý sẽ xác định xem người dùng có quyền nhận được những thông tin mà họ yêu cầu hay không.

Bộ tối ưu truy vấn thực hiện kiểm tra tốc độ. Trong khi hầu hết các hệ quản trị cơ sở dữ liệu quan hệ tối ưu cho chi phí sử dụng tài nguyên, thì MySQL tối ưu cho tốc độ, đó là lý do khiến MySQL thành công trong lĩnh vực này. Khi tìm được đường dẫn nhanh nhất để nhận thông tin, engine truy vấn sẽ thực thi truy vấn này.

Thành phần bộ nhớ đóng vai trò đảm bảo rằng quá trình xử lý truy vấn được thành công trong "query cache" (bộ đệm truy vấn). Bộ đệm truy vấn bảo trì các yêu cầu truy vấn gần đây trong không gian lưu trữ để tiết kiệm thời gian và tài nguyên cho các truy vấn tương tự sau này. Nếu một truy vấn tương tự được yêu cầu, nó sẽ lấy kết quả trong bộ đệm truy vấn, kết quả xử lý truy vấn sẽ nhanh hơn, do nó không cần phải phân tích lại cú pháp câu truy vấn. Tiếp theo, bộ đệm truy vấn lấy kết quả lần trước đó để trả lại người dùng.

Bộ quản lý giao dịch (Transaction Manager)

Một giao dịch MySQL là một tập các truy vấn được xử lý tương tự như một tiến trình đơn lẻ. Trình quản lý giao dịch làm nhiệm vụ duy trì sự đồng bộ cho toàn bộ cơ sở dữ liệu. Điều đó có nghĩa là trình quản lý giao dịch phải

chắc chắn rằng việc xử lý đồng bộ dữ liệu sẽ không làm cho dữ liệu bị lỗi hoặc trở nên không tin cậy. Trình quản lý giao dịch chịu trách nhiệm cho việc tránh và xử lý các deadlock và lỗi dữ liệu bằng cách khởi tạo câu lệnh COMMIT và ROLLBACK trên máy chủ. InnoDB và BDB là những engine lưu trữ đã được thêm vào MySQL để quản lý các giao dịch.

Có những môi trường cơ sở dữ liệu mang tính giao dịch, một số khác lại không, điều này phụ thuộc vào nhu cầu của tổ chức. MySQL là môi trường giao dịch an toàn, tuân thủ bốn nguyên tắc - ACID. Để vượt qua kiểm tra ACID, một giao dịch phải duy trì các tính chất nguyên tử, nhất quán, phân tách và bền vững. Quản trị viên MySQL có thể chọn môi trường có hoặc không có cơ sở dữ liệu giao dịch. Bảng sau miêu tả sự tuân thủ chuẩn ACID của MySQL.

Đặc trưng của chuẩn ACID	Miêu tả	Ví dụ về sự tuân thủ
Atomicity (tính nguyên tử)	Các câu lệnh SQL vận hành cùng nhau như một nhóm thực thể hoặc như một thực thể không thể phân tách.	Ví dụ các lệnh BEGIN, COMMIT, UNDO, ROLLBACK.
Consistency (tính nhất quán)	Các giao dịch không ảnh hưởng tới trạng thái của cơ sở dữ liệu; nhất quán các phần dữ liệu còn lại cho dù giao dịch thành công hay thất bại.	MySQL sử dụng các tập tin nhật ký (bản ghi nhị phân) để ghi lại tất cả các thay đổi tới cơ sở dữ liệu cũng như giúp phục hồi khi bị lỗi; câu lệnh ROLLBACK được sử dụng nếu cần thiết.
Isolation (sự phân tách)	Các giao dịch này chạy phân tách khỏi các	MyISAM cho phép khóa để tránh bị lỗi dữ

	giao dịch khác và không thể hiển thị cho đến khi được ủy thác.	liệu và không cho xem dữ liệu.
Durability (tính bền vững)	Các giao dịch vẫn được duy trì cho dù hệ thống bị lỗi.	Các bản ghi nhị phân có thể khôi phục lại.

Bảng 3.4 Tuân thủ ACID của MySQL

Bộ quản lý lưu trữ (Storage Manager)

Tương tự Oracle, MySQL lưu trữ dữ liệu của nó trong các file lưu trữ phụ (/var/lib/mysql). Bộ quản lý lưu trữ chịu trách nhiệm trong việc xử lý lưu trữ và nhận dữ liệu trong toàn bộ cơ sở dữ liệu. Giống như Oracle, hầu hết các hoạt động làm việc với bộ nhớ chính (loại bộ nhớ có tốc độ nhanh nhất) và bộ đệm để tốc độ tải nhanh. Quản lý lưu trữ trong MySQL gồm ba tầng xử lý bao gồm: quản lý tài nguyên, quản lý vùng đệm, và quản lý lưu trữ.

Quản lý tài nguyên (Resource manager)	Nhận các yêu cầu để khởi tạo bộ nhớ bằng cách thực thi và tìm kiếm trong quản lý bộ đệm nhằm đáp ứng yêu cầu này.
Quản lý bộ đệm (Buffer manager)	<ul style="list-style-type: none"> • Phân bổ bộ nhớ và xác định số lượng các buffer để phân chia tài nguyên. • Sử dụng trình quản lý lưu trữ để dàn xếp với bộ lưu trữ phụ.
Quản lý lưu trữ (Storage manager)	Yêu cầu dữ liệu từ bộ lưu trữ phụ và chuyển qua trình quản lý bộ đệm.

Bảng 3.5 Quản lý lưu trữ trong MySQL

Engin lưu trữ (The Storage Engine)

Engin lưu trữ là những thành phần của kiến trúc cơ sở dữ liệu quan hệ MySQL thực hiện đọc và ghi dữ liệu giữa cơ sở dữ liệu với các dịch vụ. Các engin lưu trữ có thể tách rời MySQL, khác với những phần còn lại của hệ quản trị cơ sở dữ liệu quan hệ MySQL, bởi vì trong MySQL, các quản trị có thể chọn lọc và chọn những engin lưu trữ mà họ muốn cho các bảng hoặc ứng dụng nào đó.

Việc có thêm những tùy chọn này giúp quản trị viên có thêm nhiều sự kiểm soát tin cậy và bảo mật với dữ liệu. Do đó, ta nên quyết định các tùy chọn này sớm ngay từ khâu thiết kế để tránh những phát sinh trong quá trình phát triển sau này. Một số engin lưu trữ có sẵn trong MySQL.

MyISAM: Engin lưu trữ mặc định của MySQL, hỗ trợ số hành động và 256 TB dữ liệu, MyISAM là một trong những engin lưu trữ hay sử dụng và được hỗ trợ trong tất cả các kiểu cấu hình MySQL. Phù hợp với cả máy chủ kiểu OLTP và DSS, MyISAM mang đến các thành phần engin kết hợp và tìm kiếm nhanh. Engin này có thể cung cấp môi trường không có tính giao dịch và hỗ trợ khóa bảng.

InnoDB: Một tùy chọn lưu trữ phổ biến khác, được thiết kế dành riêng cho OLTP. InnoDB là engin tuân thủ giao dịch cung cấp hiệu năng cao cho các giao dịch ngắn hạn thường xuyên. InnoDB gồm có COMMIT, ROLLBACK, và LOCK mức hàng.

Memory: Thường được gọi là HEAP, engin này lưu trữ mọi thứ trong RAM, do vậy ta có thể truy xuất dữ liệu cố định rất nhanh. Engin này phù hợp với môi trường DSS và các dữ liệu đã được gộp.

Archive: Engin này phù hợp cho việc lưu trữ và nhận dữ liệu. Engin lưu trữ này chỉ hỗ trợ các truy vấn INSERT, SELECT khi mà việc quét lại toàn bộ bảng là cần thiết cho mỗi câu lệnh SELECT.

Federated: Cho phép phân tách máy chủ MySQL thành các liên kết, tham chiếu tới các máy chủ từ xa cho tất cả các tác vụ. Engin này lưu trữ mọi thứ

trên máy chủ ở xa, do vậy nó rất phù hợp với môi trường phân tán, nơi mà dữ liệu không cần phải lưu trữ cục bộ.

Comma-separated values (CSV): lưu trữ dữ liệu bằng các dấu phân cách trong file văn bản (text). Thích hợp với các loại bản ghi nhật ký, engin này cung cấp sự dễ dàng chuyển đổi dữ liệu giữa các ứng dụng.

Black Hole Engine: Chấp nhận dữ liệu, nhưng không có tiện ích lưu trữ, vì vậy nó hủy bỏ các lệnh INSERT và không nhận lại dữ liệu. Nó hữu ích trong môi trường phân tán hoặc trong các cấu hình bản sao.

The Falcon Engine: Được thiết kế cho các bộ vi xử lý 64-bit, và bộ nhớ dung lượng lớn, nhưng đủ linh động để làm việc trong các môi trường nhỏ hơn. Falcon là engin an toàn giao dịch, có khả năng xử lý các giao dịch ngắn và thường xuyên.

3.2.1.3 Kiến trúc của SQL Server

Engin quan hệ (Rational Engine)	Quản lý kết nối			
	Cơ sở dữ liệu			
	Lưu trữ		Xử lý truy vấn	
	Quản lý bộ đệm	Quản lý File	Phân tích cú pháp	Tối ưu
	Các tiến trình và các luồng			
	Buffer Pool	Lập lịch thread	Quản lý bộ nhớ	Worker Threads

Hình 3.14 Kiến trúc Microsoft SQL Server

Các Engin và kiến trúc

Kiến trúc của Microsoft SQL Server được phát triển cho nền tảng Windows. Cấu trúc chính của SQL Server gồm có các hoạt động kết nối, engin quan hệ, xử lý truy vấn, engin lưu trữ, xử lý giao dịch và các file vật lý.

Các máy trạm kết nối đến máy chủ sử dụng luồng dữ liệu dạng bảng TDS (Tabular Data Stream), một định dạng của riêng của Microsoft. TDS là giao

thúc định nghĩa bởi Microsoft, đặc tả cách máy chủ SQL và máy trạm có thể tương tác với nhau. TDS miêu tả kiểu dữ liệu, cũng như cách dữ liệu có thể di chuyển. TDS thường được đóng gói bên trong một giao thức khác như TCP/IP, vì vậy các máy trạm muốn chuyển dữ liệu từ máy chủ SQL có thể sử dụng TCP/IP, cũng như các giao thức khác mà TDS được đóng gói. Kiến trúc của máy chủ cơ sở dữ liệu quan hệ SQL có thể chia thành hai thành phần chính: engin quan hệ và engin lưu trữ.

Engin quan hệ (Relational Engine): Engin quan hệ chịu trách nhiệm chính xử lý truy vấn và nhận lại dữ liệu. Nhiệm vụ mà engin quan hệ hoàn thành được khởi tạo bởi các truy vấn của người dùng và bao gồm việc phân tích cú pháp câu lệnh SQL, tối ưu quá trình thực thi, nhận lại kết quả cho người dùng.

Engin lưu trữ (Storage Engine): Engin lưu trữ chịu trách nhiệm chính để đảm bảo quản lý hiệu quả các file, bộ nhớ, phục hồi, ghi nhật ký và các giao dịch. Engin lưu trữ đảm bảo rằng lượng bộ nhớ có sẵn được phân bổ thích hợp, các file được tối ưu hiệu quả, và được lưu trữ, sao lưu khi cần thiết, dữ liệu và các cấu trúc có thể được phục hồi một cách tin cậy nếu cần thiết.

Kiến trúc vật lý

Mọi hệ quản trị cơ sở dữ liệu đều cần một nơi để lưu trữ dữ liệu. SQL sử dụng 3 loại file khác nhau để lưu trữ dữ liệu: primary data file, secondary data file, log file. Mỗi kiểu cài đặt máy chủ SQL yêu cầu ít nhất một file primary và một file log.

Primary data file (file dữ liệu chính): File dữ liệu chính của cơ sở dữ liệu SQL Server tham chiếu đến tất cả các file dữ liệu secondary, và là file gốc của toàn bộ cơ sở dữ liệu. Phải có ít nhất một file primary khi cài đặt cơ sở dữ liệu. Phần mở rộng của các file này là .mdf.

Secondary data file (file dữ liệu phụ): Đây là một tùy chọn file dữ liệu trong cơ sở dữ liệu SQL Server, các file dữ liệu không phải primary file thì là secondary file. Phần mở rộng của các file này là .ndf.

Log file: File lưu trữ thông tin về các giao dịch trong cơ sở dữ liệu được sử dụng để sao lưu và phục hồi. Phải có ít nhất một file bản ghi trong cơ sở dữ liệu. Phần mở rộng của file này là .ldf.

Trong một máy chủ SQL các file dữ liệu có thể kết hợp, gộp lại đặt thành một nhóm gọi là "nhóm file" (*filegroup*). Việc xây dựng các filegroup cho phép các quản trị viên tạo các file secondary trong cùng một nhóm để quản lý tốt hơn. Một filegroup là một tập gồm một hay nhiều file vật lý trong máy chủ cơ sở dữ liệu SQL. Chỉ có các file dữ liệu mới có thể tồn tại trong cùng filegroup, các file log phải nằm riêng rẽ. Có 2 loại filegroup:

Primary filegroup (nhóm file chính): Tập các file của hệ thống máy chủ SQL, bao gồm cả các file primary data.

User-defined filegroup (nhóm file người dùng tự định nghĩa): Một tập các file tạo bởi người dùng.

Các filegroup có thể được sử dụng để quản lý cơ sở dữ liệu thành các đơn vị tài nguyên lôgic. Cách sử dụng các filegroup phụ thuộc vào môi trường và dữ liệu được lưu trữ trong đó, tổ chức các filegroup không hiệu quả có thể gây nên những xung đột trong quá trình sao lưu và phục hồi cơ sở dữ liệu. Một cơ sở dữ liệu chỉ có một filegroup.

Quản lý bộ nhớ

Một trong những đặc trưng đáng chú ý nhất của kiến trúc bộ nhớ SQL Server là nó có thể phân phối động bộ nhớ mà không cần quản trị viên can thiệp vào. Mặc định, mỗi instance của SQL Server có được bộ nhớ động cần thiết mà không gây thiếu hụt cho hệ thống. Mục tiêu của cơ sở dữ liệu, như với MySQL và Oracle là hạn chế số lần phải truy nhập thiết bị lưu trữ secondary. Ô cứng là thiết bị lưu trữ chậm nhất và có giá rẻ nhất. Bằng cách giảm thiểu số lần đọc và ghi với chúng, hiệu suất của máy chủ sẽ được tăng lên.

Bộ nhớ ảo đóng một vai trò quan trọng trong quá trình quản lý bộ nhớ. Bộ nhớ ảo là một kỹ thuật mở rộng lượng bộ nhớ có sẵn bằng cách sử dụng các đơn vị lưu trữ khác, không phải bộ nhớ trong, để lưu trữ thông tin từ một thực

thể theo cách mà các dữ liệu đó được lưu trong bộ nhớ. Các đơn vị lưu trữ sửa đổi được dùng để chuyển đổi hoặc tráo đổi dữ liệu từ thiết bị lưu trữ này sang thiết bị lưu trữ khác được gọi là các "trang". Trang là đơn vị lưu trữ chính trong SQL Server. Các trang thường được tráo đổi từ bộ nhớ trong (RAM) tới bộ nhớ phụ (đĩa cứng) để xử lý quá tải và để đảm bảo ý tưởng tối ưu lưu trữ. Vùng không gian tráo đổi được dành riêng cho các trang được gọi là pagefile. Không gian địa chỉ ảo hoàn toàn là vùng nhớ ảo được dành riêng cho chương trình.

Mục tiêu là sử dụng tối đa bộ nhớ chính cho cơ sở dữ liệu mà không vượt quá lượng bộ nhớ hiện có trên máy chủ. Để thực hiện mục tiêu này, SQL duy trì một bộ đệm trong bộ nhớ để xử lý các trang đã được đọc trước đó từ cơ sở dữ liệu. Như được định từ trước đó, khi lớn hơn kích thước của vùng đệm, thì hệ thống sẽ đọc, ghi dữ liệu thưa xuống ổ cứng. Nếu bộ đệm sử dụng quá nhiều bộ nhớ chính, kết quả là hệ điều hành phải cài đặt bộ nhớ ảo bằng cách hoán đổi giữa bộ nhớ với pagefile được dành riêng. Quá trình này sử dụng ít tài nguyên hơn và tăng hiệu năng.

Có 2 thành phần chính của kiến trúc bộ nhớ ảo của SQL Server: vùng nhớ (memory pool) và mã thực thi (executable code).

Memory pool	Executable Code
<ul style="list-style-type: none"> • Connection Context • SQL Server • System-Level Data Structures • Buffer Cache 	<ul style="list-style-type: none"> • SQL Server code • Server Net-Library DLLs • Open services code • Extend stored procedures • OLE Automation • Object code • Distributed Query OLE DB provider DLLs

Mã thực thi: là những engin cơ bản bên trong máy chủ SQL. Nó bao gồm các file thực thi (exe), và các file thư viện liên kết động (dll) của chính máy chủ. Mã thực thi bao gồm các thành phần:

- SQL Server code
- Server Net-Library DLLs
- Open services code
- Extend stored procedures
- OLE Automation
- Object code
- Distributed Query OLE DB provider DLLs

Vùng nhớ (Memory pool): Vùng nhớ là tổng dung lượng bộ nhớ hiện có cho một instance của SQL Server. Gần như mọi thứ trong một instance của SQL Server đều sử dụng vùng nhớ. Kích cỡ của vùng nhớ phụ thuộc vào số lượng các instance và các ứng dụng đang chạy. Sự thay đổi các thành phần trong một instance của SQL Server là rất biến động.

Vùng nhớ có thể dao động giữa giá trị nhỏ nhất và lớn nhất đã được thiết lập, trong khi nó cố gắng đáp ứng yêu cầu của các ứng dụng và các instance, và duy trì mức dung lượng bộ nhớ ảo chỉ dưới dung lượng bộ nhớ thật từ 4-10 MB.

Một instance máy chủ bắt đầu với lượng bộ nhớ mà nó cần, sau đó nó thêm hoặc xóa bộ nhớ bằng cách xác định số lượng các ứng dụng, các instance, và các yêu cầu của người dùng. Vùng nhớ không bao giờ vượt quá dung lượng bộ nhớ máy chủ tối đa được chỉ định. Năm thành phần chính trong vùng nhớ được phân bổ bộ nhớ:

Các cấu trúc dữ liệu mức hệ thống: lưu trữ tất cả các dữ liệu toàn cục tới instance.

Bộ đệm buffer: lưu các trang dữ liệu để tránh việc đọc, ghi xuống ổ cứng.

Bộ đệm thủ tục: lưu các kế hoạch thực thi cho tất cả các câu lệnh SQL chạy hiện tại và gần đây.

Bộ đệm nhật ký: lưu các trang thực hiện việc đọc, ghi nhật ký.

Ngữ cảnh kết nối: lưu bản ghi của trạng thái kết nối hiện tại.

Trình quản lý bộ đệm

Quản lý bộ đệm là sự sống còn đối với hiệu năng của máy chủ SQL Server. Bộ đệm là nơi nằm bên trong bộ nhớ vật lý, nơi dữ liệu được lưu trữ, để tối thiểu hóa số lần đọc, ghi tới bộ nhớ thứ cấp. Có hai thành phần quan trọng trong trình quản lý bộ đệm. Quản lý bộ đệm (buffer manager) là một phần của SQL Server phản hồi lại cho các trang truy nhập dữ liệu và cập nhật cơ sở dữ liệu. Vùng đệm (buffer pool) hoặc bộ đệm (buffer cache) là nơi mà các trang dữ liệu từ cơ sở dữ liệu được lưu trữ để giảm thiểu hóa lần đọc, ghi xuống ổ cứng.

Khi SQL Server bắt đầu chạy, nó xác định kích cỡ và dành riêng bộ đệm bằng cách tính toán lượng bộ nhớ vật lý còn dư. Vùng không gian được dành riêng cho bộ đệm được gọi là "đích nhớ" (memory target). Bộ đệm này chỉ cần thiết cho instance hiện tại.

Khi có một yêu cầu tới dữ liệu (sau quá trình xử lý truy vấn), bộ quản lý bộ đệm sẽ truy nhập tới file cơ sở dữ liệu được lưu trên ổ cứng để nhận dữ liệu yêu cầu. Dữ liệu được đưa vào trong các trang bộ đệm, và từ đây dữ liệu được đọc. Nếu có sự thay đổi tác động tới dữ liệu, trong khi dữ liệu đang nằm trong bộ đệm, nó được coi như là "dirty". Một trang dirty là một trang đã bị chỉnh sửa trong bộ đệm, nhưng chưa ghi trở lại cơ sở dữ liệu. Một trang dirty có thể trải qua vài lần chỉnh sửa trước khi nó được ghi trở lại cơ sở dữ liệu.

Với mỗi lần chỉnh sửa, một hàng chứa giao dịch được thêm vào bộ đệm nhật ký để đảm bảo tính toàn vẹn. Khi một giao dịch được ghi lại, và trang dirty đã dừng trước khi được tham chiếu (bởi người dùng hoặc câu lệnh SQL), thì bộ đệm làm sạch các dữ liệu rồi ghi ngược trở lại cơ sở dữ liệu.

3.2.2 Một số lỗ hổng trong các hệ quản trị

Các hệ quản trị cơ sở dữ liệu dù được cho là hệ quản trị mạnh nhưng vẫn chứa rất nhiều điểm yếu và lỗ hổng bên trong nó. Vấn đề này tưởng chừng như thuộc về trách nhiệm của các nhà cung cấp nhưng lại ảnh hưởng trực tiếp đến quyền lợi và sự an toàn của thông tin đối với người sử dụng. Chính vì

vậy, người sử dụng không nên coi nhẹ vấn đề này, họ cần thường xuyên cập nhật các bản vá mà các nhà cung cấp đã công bố để nâng cao chất lượng cũng như độ an toàn khi sử dụng các hệ quản trị này. Trong phần này, chúng ta tìm hiểu một số những lỗ hổng bảo mật mà các nhà cung cấp đã công bố đối với các phiên bản của Oracle, MySQL và SQL Server.

3.2.2.1 Một số lỗ hổng trong Oracle

Hãng Oracle khuyến cáo các khách hàng nên áp dụng các bản vá một cách thường xuyên nhất. Dưới đây là tập các bản vá cho các lỗ hổng bảo mật nghiêm trọng trong các sản phẩm của Oracle.

Một số bản vá lỗi đã được phát hành bởi Oracle:

Bản vá	Phiên bản áp dụng	Các lỗ hổng bảo mật
Critical Patch Update - Tháng tư, 2012	11.5.10 – 12.1.x	- Phát hành kiến trúc bảo mật Oracle E-Business Suite
Critical Patch Update - Tháng 7, 2011	11.5.10 – 12.1.x	- Phát hành cấu hình bảo mật Oracle E-Business Suite
Critical Patch Update - Tháng 10, 2010	11.5.10 – 12.1.x	- 2 điểm yếu bảo mật trong bộ Oracle E-Business Suite
Critical Patch Update - Tháng 7, 2008	Oracle 11g 11.5.8 – 12.0.x	- 2 điểm yếu trong xác thực của Oracle RDBMS - 2 lỗ hổng trong bộ Oracle E-Business Suite
Critical Patch Update - Tháng 4, 2008	12.0.x 11.5.7 – 11.5.10	- 8 lỗ hổng, SQL Injection, XSS, lộ thông tin, v.v...
Critical Patch Update - Tháng 7, 2007	12.0.x 11.5.1 – 11.5.10	- 11 lỗ hổng, SQL Injection, XSS, lộ thông tin, v.v...
Critical Patch Update -	11.0.x, 11.5.1 –	- Phát hành cấu hình mặc

Tháng 10, 2005	11.5.10	định
Critical Patch Update - Tháng 7, 2005	11.0.x, 11.5.1 – 11.5.10	- Lỗ hổng SQL Injection và lộ thông tin
Critical Patch Update - Tháng 4, 2005	11.0.x, 11.5.1 – 11.5.10	- Lỗ hổng SQL Injection và lộ thông tin
Critical Patch Update - Tháng 1, 2005	11.0.x, 11.5.1 – 11.5.10	- Lỗ hổng SQL Injection
Oracle Security Alert #68	Oracle 8i, 9i, 10g	- Buffer overflows - Lộ thông tin Listener
Oracle Security Alert #67	11.0.x, 11.5.1 – 11.5.8	- 10 lỗ hổng SQL injection
Oracle Security Alert #56	11.0.x, 11.5.1 – 11.5.8	- Buffer overflow trong FNDWRR.exe
Oracle Security Alert #55	11.5.1 – 11.5.8	- Các lỗ hổng trong AOL/J Setup Test - Lấy các thông tin nhạy cảm (session hợp lệ)
Oracle Security Alert #53	10.7, 11.0.x 11.5.1 – 11.5.8	- Không có xác thực trong chương trình FNDFS - Lấy mọi file từ hệ điều hành

Bảng 3.6. Các bản vá lỗi của Oracle

Tháng 4, năm 2013 hãng Oracle phát hành bản vá "Oracle Critical Patch Update Advisory - April 2013". Các sản phẩm hệ quản trị cơ sở dữ liệu được vá bởi bản vá này gồm có:

Oracle Database 11g Release 2, phiên bản 11.2.0.2, 11.2.0.3

Oracle Database 11g Release 1, phiên bản 11.1.0.7

Oracle Database 10g Release 2, phiên bản 10.2.0.4, 10.2.0.5

Oracle Application Express, phiên bản trước 4.2.1

Chi tiết về danh sách các sản phẩm được vá bởi bản vá này, bạn đọc có thể tham khảo tại website của hãng:

<http://www.oracle.com/technetwork/topics/security/cpuapr2013-1899555.html>

Ngoài ra, để tìm hiểu về các lỗ hổng bảo mật đã công bố của Oracle ta có thể tìm hiểu qua các kho lưu trữ CVE (common identifiers for publicly known information security vulnerabilities). Sau đây là một số CVE điển hình:

CVE-2012-1675

- Loại lỗ hổng: Exec code
- Ngày công bố: 2012-05-08
- Ngày cập nhật: 2012-07-21
- Điểm đánh giá: 7.5/10
- Truy nhập: Remote
- Độ phức tạp: thấp
- Xác thực: không

Mô tả: lỗ hổng này xuất hiện trong bộ TNS Listener, được sử dụng trong các sản phẩm Oracle Database 11g 11.1.0.7, 11.2.0.2, 11.2.0.3, và 10g 10.2.0.3, 10.2.0.4, 10.2.0.5, cũng như được sử dụng trong Oracle Fusion Middleware, Enterprise Manager, E-Business Suite, và có thể trong các sản phẩm khác. Lỗ hổng này cho phép kẻ tấn công từ xa thực hiện các câu lệnh cơ sở dữ liệu tùy ý bằng cách đăng ký từ xa một instance hoặc service name đã tồn tại, sau đó thực hiện tấn công người xen giữa (man-in-the-middle - MITM) để đạt kết nối cơ sở dữ liệu, đây còn gọi là tấn công đầu độc TNS (TNS poison).

CVE-2009-0985

- Loại lỗ hổng: chưa được miêu tả
- Ngày công bố: 2009-04-15
- Ngày cập nhật: 2012-10-22
- Điểm đánh giá: 7.1/10
- Truy nhập: Remote

- Độ phức tạp: cao
- Xác thực: Single system

Mô tả: lỗ hổng này chưa được miêu tả trong thành phần lỗi RDBMS của Oracle Database 10.1.0.5, 10.2.0.4, và 11.1.0.6. Lỗ hổng này cho phép kẻ tấn công xác thực từ xa với vai trò IMP_FULL_DATABASE gây ảnh hưởng tới tính bí mật, toàn vẹn và sẵn sàng.

CVE-2009-0972

- Loại lỗ hổng: chưa được miêu tả
- Ngày công bố: 2009-04-15
- Ngày cập nhật: 2012-10-22
- Điểm đánh giá: 6.5/10
- Truy nhập: Remote
- Độ phức tạp: thấp
- Xác thực: Single system

Mô tả: lỗ hổng chưa được miêu tả trong thành phần Workspace Manager của Oracle Database 11.1.0.6, 11.1.0.7, 10.2.0.3, 10.2.0.4, 10.1.0.5, 9.2.0.8, và 9.2.0.8DV. Lỗ hổng này cho phép kẻ tấn công từ xa có thể gây ảnh hưởng tới tính bí mật, toàn vẹn, sẵn sàng.

3.2.2.2 Một số lỗ hổng trong MySQL

Bạn đọc có thể tham khảo danh sách các lỗ hổng đã được công bố của MySQL tại một số địa chỉ sau:

- 1.http://www.cvedetails.com/vulnerability-list/vendor_id-185/cvsscoremin-9/cvsscoremax-/Mysql.html
- 2.http://www.cvedetails.com/vulnerability-list/vendor_id-185/product_id-316/version_id-102491/Mysql-Mysql-5.5.3.html

Sau đây là một số mô tả về một vài lỗ hổng trong MySQL đã được công bố:

CVE-2013-2392

- Ngày công bố: 2013-04-17
- Ngày cập nhật: 2013-06-04

- Điểm đánh giá: 4.0
- Truy nhập: Remote
- Độ phức tạp: Thấp
- Xác thực: Single system
- Ảnh hưởng tính bí mật: Không
- Ảnh hưởng tính toàn vẹn: Không
- Ảnh hưởng tính sẵn sàng: Một phần

Mô tả: lỗ hổng này chưa được miêu tả trong và trước các phiên bản MySQL 5.1.68, 5.5.30, 5.6.10, nó cho phép kẻ tấn công từ xa gây ảnh hưởng tới tính sẵn sàng liên quan đến bộ tối ưu máy chủ (Server Optimizer).

CVE-2013-2391

- Ngày công bố: 2013-04-17
- Ngày cập nhật: 2013-06-04
- Điểm đánh giá: 3.0
- Truy nhập: Local
- Độ phức tạp: Trung bình
- Xác thực: Single system
- Ảnh hưởng tính bí mật: Một phần
- Ảnh hưởng tính toàn vẹn: Một phần
- Ảnh hưởng tính sẵn sàng: Không

Mô tả: lỗ hổng này chưa được miêu tả trong và trước các phiên bản MySQL 5.1.68, 5.5.30, 5.6.10, nó cho phép người dùng cục bộ gây ảnh hưởng tới tính bí mật, tính toàn vẹn liên quan đến việc cài đặt máy chủ (Server Install).

CVE-2012-3163

- Ngày công bố: 2012-10-16
- Ngày cập nhật: 2013-02-07
- Điểm đánh giá: 9.0
- Truy nhập: Remote
- Độ phức tạp: Thấp
- Xác thực: Single system

- Ảnh hưởng tính bí mật: Toàn bộ
- Ảnh hưởng tính toàn vẹn: Toàn bộ
- Ảnh hưởng tính sẵn sàng: Toàn bộ

Mô tả: lỗ hổng này chưa được miêu tả trong thành phần MySQL Server trong và trước các phiên bản của MySQL 5.1.64 và 5.5.26, nó cho phép kẻ tấn công từ xa gây ảnh hưởng tới tính bí mật, toàn vẹn, sẵn sàng liên quan đến lược đồ thông tin (Information Schema).

CVE-2012-2750

- Ngày công bố: 2012-08-16
- Ngày cập nhật: 2012-08-17
- Điểm đánh giá: 10.0
- Truy nhập: Remote
- Độ phức tạp: Thấp
- Xác thực: Không yêu cầu
- Ảnh hưởng tính bí mật: Toàn bộ
- Ảnh hưởng tính toàn vẹn: Toàn bộ
- Ảnh hưởng tính sẵn sàng: Toàn bộ

Mô tả: lỗ hổng chưa được miêu tả trong các phiên bản MySQL 5.5.x trước 5.5.23, tấn công liên quan đến "Security Fix", hay Bug #59533.

3.2.2.3 Một số lỗ hổng trong SQL Server

Dưới đây là một số lỗ hổng cơ bản của hệ quản trị SQL Server:

CVE-2012-1856

Mô tả: Là lỗ hổng trong "Windows Common Controls - MSCOMCTL.OCX" có thể cho phép thực thi mã điều khiển từ xa. Lỗi này gây ảnh hưởng tới một số các sản phẩm cài đặt mặc định Windows common controls, các sản phẩm cơ sở dữ liệu bị ảnh hưởng gồm có:

- Microsoft SQL Server 2000 Analysis Services.
- Microsoft SQL Server 2000 (ngoại trừ phiên bản dựa trên Itanium).

- Microsoft SQL Server 2005 (ngoại trừ Microsoft SQL Server 2005 Express Edition, nhưng bao gồm Microsoft SQL Server 2005 Express Edition đi kèm với Advanced Services).
- Microsoft SQL Server 2008.
- Microsoft SQL Server 2008 R2.

CVE-2012-2552

Mô tả: lỗ hổng trong Microsoft SQL Server trên các hệ thống đang chạy SQL Server Reporting Services (SSRS). Đây là lỗ hổng XSS, có thể cho phép leo thang đặc quyền, cho phép kẻ tấn công thực hiện câu lệnh tùy ý trên phía SSRS trong ngữ cảnh của người dùng đích. Kẻ tấn công có thể khai thác lỗ hổng này bằng cách gửi cho người dùng một đường link được ngụy trang khéo léo và dụ người dùng bấm vào link đó. Lỗ hổng này ảnh hưởng tới các sản phẩm:

- Microsoft SQL Server 2000 Reporting Services Service Pack 2.
- Mọi hệ thống đang chạy SQL Server Reporting Services (SSRS) trên Microsoft SQL Server 2005 Service Pack 4.
- Microsoft SQL Server 2008 Service Pack 2.
- Microsoft SQL Server 2008 Service Pack 3.
- Microsoft SQL Server 2008 R2 Service Pack 1.
- Microsoft SQL Server 2012.

Ngoài ra, bạn đọc có thể tham khảo thêm một số lỗ hổng khác của MySQL đã công bố tại một số địa chỉ sau:

- <http://www.microsoft.com/en-us/download/details.aspx?id=26482>
Security Update for SQL Server 2005 Service Pack 3 (KB2494113).
- http://www.cvedetails.com/vulnerability-list/vendor_id-26/product_id-251/version_id-82478/Microsoft-Sql-Server-2005.html
- <http://technet.microsoft.com/en-us/security/bulletin/ms12-apr>
- <http://technet.microsoft.com/en-us/security/bulletin/ms13-apr>

3.3 CÁC VẤN ĐỀ AN TOÀN CHUNG TRONG DBMS

3.3.1 Xác thực (Authentication)

Để đảm bảo việc truy nhập tới cơ sở dữ liệu được an toàn, cần mất rất nhiều thời gian và sự nỗ lực. Chúng ta muốn chắc chắn rằng dữ liệu chỉ được trao quyền cho các cá nhân hay ứng dụng đã được thẩm định, thì cần sử dụng hợp lý nhiều lớp bảo mật kết hợp với nhau. Có hai bước chính để kiểm soát truy nhập dữ liệu là: xác thực và cấp quyền. Xác thực là quá trình xác nhận định danh của các cá nhân hay ứng dụng có yêu cầu truy nhập tới một môi trường an toàn. Sự xác nhận định danh này được hoàn thành bằng cách thẩm tra đăng nhập và các chứng nhận (credential) được tạo trong cùng một môi trường.

Quá trình đăng nhập khác với tài khoản người dùng ở chỗ, đăng nhập yêu cầu xác thực đối với môi trường, còn tài khoản người dùng được dùng để kiểm soát các hành động thực hiện đối với môi trường. Sự phân biệt này là rất rõ ràng vì có những đăng nhập mặc định được tạo trong suốt quá trình cài đặt cơ sở dữ liệu. Do vậy, có một số tài khoản người dùng mặc định cần phải được quản lý đúng đắn.

Một chứng nhận là một mẫu thông tin được dùng để kiểm tra định danh, chẳng hạn như: tên truy nhập và mật khẩu của một người dùng, mã định danh bảo mật của một ứng dụng, hoặc tên máy và địa chỉ mạng. Loại chứng nhận được sử dụng để kiểm tra định danh của của một người hoặc một ứng dụng phụ thuộc vào các yêu cầu và quá trình xác thực của từng hệ thống hoặc môi trường. Với môi trường này, tên truy nhập và mật khẩu có thể là đủ, nhưng với một môi trường khác có thể cần một số thông tin bổ sung như: địa chỉ máy và số định danh bảo mật.

Xác thực có thể được kiểm tra ở nhiều thời điểm và ở những mức độ khác nhau trong suốt quá trình đăng nhập vào hệ thống trước khi hệ thống cấp quyền cho người dùng. Ví dụ, SQL Server kiểm tra xác thực của một người dùng ở mức máy chủ bằng cách yêu cầu thiết lập kết nối tới máy chủ, và ở mức cơ sở dữ liệu bằng cách yêu cầu truy nhập tới cơ sở dữ liệu. Các ứng dụng của bên thứ ba có thể được sử dụng trong môi trường cơ sở dữ liệu

nhằm bổ sung thêm độ bảo mật trong việc xác thực người dùng. Ở đó, một số bước được bổ sung trong quá trình xác thực, chẳng hạn như: mã hóa mật khẩu để giữ một môi trường mạng an toàn.

Có ba mức xác thực thường xuyên trong môi trường cơ sở dữ liệu, đó là: mức hệ điều hành, mức cơ sở dữ liệu, và hỗ trợ của bên thứ ba. Khi được kết hợp với nhau, các mức này tạo nên một môi trường an toàn, nhưng nếu chỉ sử dụng đơn lẻ có thể mang lại những thuận lợi hoặc bất lợi lớn đối với sự an toàn của môi trường.

3.3.1.1 Xác thực mức hệ điều hành

Các chứng nhận được xác thực thông qua hệ điều hành thì phải có tài khoản trên hệ điều hành đó, và các tài khoản này được sử dụng để truy nhập hệ thống. Trong một vài trường hợp, chỉ có một bước đăng nhập hệ điều hành được sử dụng để xác thực người dùng với cơ sở dữ liệu. Điều này có nghĩa rằng, nếu người dùng có một tài khoản hệ điều hành trên hệ thống và các chứng nhận này được sử dụng thì người dùng không cần thiết phải có thêm các bước đăng nhập khác để truy nhập vào cơ sở dữ liệu. Xác thực thông qua hệ điều hành cho phép người dùng thuận tiện kết nối đến cơ sở dữ liệu mà không cần chỉ định tên người dùng và mật khẩu cơ sở dữ liệu. Loại xác thực này cũng cung cấp cho các DBA một ưu điểm, đó là khả năng quản trị tài khoản tập trung, bởi vì mọi tài khoản được đặt cùng một chỗ và mỗi cá nhân chỉ cần thiết lập chứng nhận của mình do hệ điều hành quản lý.

Tuy nhiên, với xác thực mức hệ điều hành, các máy trạm và các tài khoản người dùng phải được quản lý và giám sát thường xuyên hơn. Nếu một đối tượng bất hợp pháp lấy được các chứng nhận của người dùng để truy nhập vào hệ điều hành hoặc có thể truy nhập được vào máy trạm, thì anh ta sẽ được thừa hưởng quyền truy nhập cơ sở dữ liệu mà không cần phải có nhiều sự hiểu biết hoặc kỹ năng để truy nhập cơ sở dữ liệu từ hệ điều hành.

3.3.1.2 Xác thực mức cơ sở dữ liệu

Với xác thực mức cơ sở dữ liệu, các chứng nhận được cơ sở dữ liệu kiểm tra lần nữa. Trong tình huống này, người dùng có thể được yêu cầu truy nhập

tới một vài hệ thống trước khi đến với cơ sở dữ liệu. Có nghĩa rằng, người dùng phải lưu giữ các chứng nhận tài khoản cho những hệ thống khác. Như vậy, cùng lúc người dùng phải nhớ rất nhiều tài khoản, điều này thường dẫn đến các vấn đề làm yếu mật khẩu. Ngoài ra, việc quản trị môi trường này cũng khó khăn hơn. Các quản trị viên không những phải lưu giữ nhiều hơn một tài khoản cho mỗi cá nhân, mà các tài khoản này thường xuyên được đặt ở những khu vực riêng, cho nên việc thực hiện vết kiểm toán sẽ rất tốn công.

Tuy nhiên, việc phân tách riêng các tài khoản dành cho các hệ thống khác nhau có thể tạo ra môi trường phân đoạn và bảo mật hơn. Những người xâm phạm truy nhập bất hợp pháp tới mật khẩu người dùng cũng sẽ cần phải có thêm chứng nhận cơ sở dữ liệu để truy nhập nó.

3.3.1.3 Xác thực mạng hoặc bên thứ ba

Việc xác thực cơ sở dữ liệu có thể quản lý bằng cách sử dụng các ứng dụng bên thứ ba ở mức mạng của môi trường. Các ứng dụng bên thứ ba và các tài khoản xác thực mạng có thể được sử dụng cho truy nhập từ xa và môi trường vật lý. Người dùng không cần tạo tài khoản trên trên hệ điều hành hoặc cơ sở dữ liệu. Tuy nhiên, họ phải có tài khoản mạng hoặc tài khoản được nhận ra bởi ứng dụng bên thứ ba. Chẳng hạn, một loại xác thực liên kết đó là thẻ thông minh. Các thẻ thông minh yêu cầu người dùng thêm số PIN để xác thực; số PIN này không liên quan tới hệ điều hành và cơ sở dữ liệu.

Các giao thức bảo mật như Kerberos thường xuyên được dùng cho xác thực mạng và xác thực từ các hãng thứ ba. Các giao thức này yêu cầu nhiều hiểu biết và kinh nghiệm kỹ thuật hơn, và cũng bổ sung thêm nhiều lớp bảo mật hơn cho một môi trường cơ sở dữ liệu bất kỳ. Kerberos là giao thức xác thực được xây dựng bởi học viện kỹ thuật Massachusset (MIT), cung cấp xác thực an toàn bằng cách sử dụng mã hóa khóa đối xứng để định danh thực thể từ máy trạm đến máy chủ và ngược lại.

Hình thức xác thực mạng cũng có thể sử dụng các giao thức bảo mật khác để xác nhận định danh một người dùng. Chẳng hạn, hạ tầng khóa công khai (PKI). PKI sử dụng các khóa được mã hóa tương tự như cách Kerberos phát

hành vé TGT, nó xác nhận định danh của người yêu cầu bằng cách gán một chứng chỉ số cho người dùng để xác thực tới môi trường bảo mật. Một chứng chỉ số là file chứa thông tin định danh đối tượng, được phát hành bởi một trung tâm chứng thực tin cậy. Các cá nhân và các công ty có thể nhận được chứng chỉ số bằng cách đăng ký qua một trung tâm chứng thực tin cậy như VeriSign. Điều quan trọng là xác thực mạng hoặc xác thực bên thứ ba phải được xem xét cẩn thận, đồng thời các kỹ thuật sử dụng cho xác thực phải được nghiên cứu qua một thời gian trước khi được cài đặt. Bởi vì, xác thực mạng sử dụng các kỹ thuật mật mã phức tạp để xác thực người dùng, nhưng trong đó còn rất nhiều vấn đề cần quan tâm, nếu không có hiểu biết đầy đủ, có thể nảy sinh nhiều vấn đề nguy hiểm cho mạng.

Xác thực bên thứ ba hay xác thực liên kết có thể được kết hợp với hệ điều hành và máy chủ xác thực để tạo ra bối cảnh sử dụng an toàn cho bất cứ môi trường nào.

3.3.1.4 Các thành phần xác thực của nhà cung cấp cơ sở dữ liệu

Có vài sự khác biệt quan trọng trong cách xác thực giữa các nhà cung cấp cơ sở dữ liệu khác nhau, như: SQL Server, MySQL, Oracle.

Xác thực trong SQL Server

SQL Server hỗ trợ hai chế độ xác thực: Windows Authentication mode (chế độ xác thực Windows), và Mixed Mode Authentication (chế độ xác thực hỗn hợp). Chế độ xác thực Windows là kiểu xác thực chỉ sử dụng hệ điều hành Windows để xác thực truy nhập tới cơ sở dữ liệu. Xác thực Windows còn được gọi là xác thực tin cậy vì tính an toàn được thực thi qua hệ điều hành Windows. Đây là chế độ mặc định trong suốt quá trình cài đặt và nó là chế độ xác thực được khuyến nghị cho SQL Server. Nó bảo mật hơn các phương pháp thay thế, và sử dụng giao thức Kerberos, yêu cầu mật khẩu mạnh và thời gian hết hạn của mật khẩu.

Khi người dùng truy nhập bằng tài khoản Windows của mình, việc kiểm tra người dùng được thực hiện bởi Windows và không cần thêm các chứng nhận để truy nhập cơ sở dữ liệu. Xác thực máy chủ SQL Server bị vô hiệu hóa

khi xác thực bằng Windows được lựa chọn. Chế độ xác thực hỗn hợp là kiểu xác thực cho phép cả xác thực Windows và xác thực máy chủ SQL Server được sử dụng để truy nhập tới cơ sở dữ liệu. Vì vậy cơ sở dữ liệu chấp nhận cả đăng nhập Windows và Server. Hiển nhiên để truy nhập tới cơ sở dữ liệu sử dụng chế độ hỗn hợp, người dùng phải đáp ứng hai chứng nhận riêng rẽ. Tuy nhiên, nhiều tài khoản mang đến khó khăn cho quản trị và an toàn cơ sở dữ liệu người dùng. Chế độ này còn được gọi là kết nối không an toàn bởi vì nó không an toàn như chế độ xác thực Windows và vì các giao thức như Kerberos không được sử dụng. Chế độ xác thực hỗn hợp thích hợp cho các môi trường với hệ điều hành cũ và môi trường hỗn hợp hệ điều hành.

Xác thực trong MySQL

MySQL sử dụng giao thức xác thực để truy nhập đến máy chủ và định danh khác so với SQL Server và Oracle. Định danh của người dùng MySQL được kiểm tra bằng cách sử dụng ba mẩu thông tin:

Tên máy chủ đang chạy.

Tên truy nhập MySQL.

Mật khẩu truy nhập MySQL.

Để truy nhập cơ sở dữ liệu, định danh phải trùng khớp với các chứng nhận được lưu trữ trong cơ sở dữ liệu. Tên đăng nhập phân tách hoàn toàn với tài khoản đăng nhập hệ điều hành, và mật khẩu cũng vậy, không có mối liên hệ nào giữa mật khẩu MySQL và mật khẩu hệ điều hành. Tên máy có thể cho dưới dạng địa chỉ IP, và cũng không cần cung cấp tên máy nếu máy đó chính là máy chủ đang chạy MySQL. Có ba thành phần được lưu trữ trong bảng người dùng (host, user, password). Kết nối chỉ được cho phép nếu ba giá trị đó được kiểm tra trùng khớp.

Xác thực trong Oracle

Oracle hỗ trợ nhiều tùy chọn xác thực cho người dùng, ứng dụng và các máy, và nó cung cấp tùy chọn cấu hình hỗ trợ hầu hết các môi trường. Các máy chủ cơ sở dữ liệu, các liên kết cơ sở dữ liệu, và các mật khẩu môi trường đều có thể được sử dụng như một chứng nhận để xác thực trong Oracle 11g.

Ngoài ra, trong Oracle một vài ứng dụng được thêm vào để tăng thêm bảo mật cho hạ tầng cơ sở dữ liệu.

Một trong những dịch vụ đáng chú ý nhất được thêm vào Oracle để tăng cường thêm tính bảo mật đó là "Advanced Security". Advanced Security là một ứng dụng bảo mật toàn diện. Nó mã hóa toàn bộ thông tin chuyển qua mạng và lưu trữ trong cơ sở dữ liệu để cung cấp các xác thực mạnh và xác thực proxy quan trọng, hỗ trợ và tích hợp cùng với các phương pháp xác thực chuẩn công nghiệp (Kerberos, PKI, SSL).

3.3.1.5 Chính sách mật khẩu

Chúng ta biết rằng, mật khẩu là chìa khóa để truy nhập vào tài khoản người dùng. Hầu hết các vụ tấn công vào hệ thống xuất phát từ việc bẻ khóa hoặc lấy cắp mật khẩu của người dùng. Hơn nữa, phải tốn rất nhiều tiền của, thời gian, tài nguyên để có thể khắc phục các mật khẩu bị tổn hại. Do đó, chính sách mật khẩu cần thiết được sử dụng. Chính sách mật khẩu là việc làm có tác dụng đầu tiên để giảm rủi ro cho mật khẩu. Máy chủ dữ liệu được cấu hình với cơ sở dữ liệu trong suốt với người quản trị hệ thống.

Cơ sở dữ liệu – Thực thi chính sách mật khẩu

Phần này định nghĩa về việc cài đặt mật khẩu chung trong một môi trường quản lý dữ liệu. Nhà quản trị cơ sở dữ liệu và chuyên gia bảo mật cần thiết lập các quy tắc cho người dùng trong việc sử dụng các thiết bị và công nghệ trong công ty. Có bốn thuộc tính của mật khẩu có thể thực thi trong hầu hết các máy chủ dữ liệu, bao gồm:

Độ phức tạp: Tạo một định nghĩa yêu cầu về độ dài, kiểu chữ (số, ký tự, chữ cái...) của mật khẩu.

Số lần đăng nhập thất bại: Một mật khẩu thử quá số lần quy định có thể là dấu hiệu của việc xâm nhập bất hợp pháp. Do đó, cần giới hạn số lần đăng nhập thất bại nhằm giảm thiểu rủi ro với mật khẩu, đồng thời sử dụng chính sách khóa tạm thời với tài khoản bị nhập quá số lần quy định.

Mật khẩu quá hạn: cần một chính sách quy định thời gian sử dụng mật khẩu đối với người dùng. Nó giúp giảm thiệt hại nếu mật khẩu bị lấy cắp.

Tái sử dụng mật khẩu: quy định một mật khẩu có được sử dụng lại hay không.

Thiết lập chính sách mật khẩu

Việc thiết lập chính sách mật khẩu bao gồm các phần sử dụng phù hợp giữa người dùng và công ty. Thiết lập chính sách mật khẩu cần linh hoạt để thực thi một cách nhất quán với mỗi mức của công ty và đầy đủ tính nghiêm ngặt để đảm bảo người dùng tuân thủ chính sách.

3.3.2 Cấp quyền (Authorization)

Sau khi hệ thống đã xác nhận danh tính hay đăng nhập của một người dùng thì một tập các điều khoản được đưa ra nhằm xác định xem đối tượng có được hay không được truy nhập vào cơ sở dữ liệu. Cấp quyền là quá trình đảm bảo rằng những cá nhân hoặc các ứng dụng yêu cầu truy nhập vào một môi trường hoặc một đối tượng trong môi trường có sự cho phép hay không.

Quản lý tài khoản người dùng

Quản lý tài khoản đúng cách rất quan trọng trong việc kiểm soát an toàn và truy nhập cơ sở dữ liệu. Các hoạt động phổ biến nhất mà người quản trị cơ sở dữ liệu cần thực hiện trên một cơ sở dữ liệu, đó là những vấn đề liên quan đến việc quản lý người dùng. Ở mức tối thiểu trong một cơ sở dữ liệu, một quản trị viên phải biết làm thế nào để thêm, gỡ bỏ, và gán các đặc quyền cho người sử dụng trong môi trường của mình.

Tài khoản mặc định

Trong hầu hết các cơ sở dữ liệu, tài khoản người dùng mặc định được tạo ra, trong đó tên người dùng truy nhập được xác định trước. Hầu hết các người dùng mặc định được tạo ra trong quá trình cài đặt là người dùng hệ thống hoặc người quản trị tài khoản, nắm giữ quyền truy nhập cao nhất vào bất kỳ vị trí nào trong cơ sở dữ liệu. Thông tin về các tài khoản này như: mật khẩu mặc định, tên người dùng, quyền và đặc quyền, và khả năng tiếp cận tài khoản, có thể dễ dàng tìm thấy bằng cách thực hiện một tìm kiếm trực tuyến đơn giản.

Thêm và loại bỏ người sử dụng

Thêm người dùng vào một cơ sở dữ liệu là một quá trình khá đơn giản nếu người quản trị cơ sở dữ liệu đã có kế hoạch thích hợp và chuẩn bị trước danh sách quyền và khả năng tiếp cận dành cho người dùng đó. Trong việc tạo ra một tài khoản người dùng, quyền bảo mật và truy nhập cũng được áp dụng. Người dùng mới phải thay đổi mật khẩu mặc định trước khi truy nhập.

Việc loại bỏ một người dùng có thể phức tạp hơn với người quản trị. Bởi vì, việc loại bỏ một tài khoản người dùng khỏi hệ thống liên quan đến tất cả các đối tượng mà người dùng này sở hữu, đồng thời tùy thuộc vào vai trò của người dùng này trong công ty mà việc loại bỏ có thể là một rủi ro. Do vậy, trước khi gỡ bỏ bất kỳ người dùng nào ra khỏi cơ sở dữ liệu, người quản trị nên thực hiện kiểm kê cẩn thận các đối tượng mà người dùng đã tạo ra và sao lưu các tài khoản đó.

Đặc quyền người dùng

Đặc quyền người dùng có thể được quản lý, từ chối, hoặc thu hồi trong một cơ sở dữ liệu

- Cấp một đặc quyền (grant Privilege) – Là hành động cung cấp, trao cho một người dùng các quyền truy nhập hay quyền thực hiện một hành động trong một cơ sở dữ liệu.
- Từ chối đặc quyền (Deny Privilege)– là hành động từ bỏ các quyền truy nhập hoặc quyền thực hiện một hành động trong một cơ sở dữ liệu của một người dùng.
- Thu hồi đặc quyền (Revoke Privilege) – là hành động lấy đi hoặc thu hồi một đặc quyền đã cấp hoặc bị từ chối trước đây.

Roles

Những đặc quyền liên quan đến nhau có thể được kết hợp lại để tạo ra một role, để quản lý tập trung một nhóm đối tượng hoặc người dùng trong một cơ sở dữ liệu. Một role là một tập hợp các đặc quyền liên quan được kết

hợp để cung cấp một đơn vị tập trung mà từ đó có thể quản lý những người dùng hay đối tượng giống nhau trong cơ sở dữ liệu.

Roles được tạo ra cho người sử dụng, các đối tượng và các ứng dụng giống nhau và cung cấp nhiều thuận lợi trong việc quản lý cơ sở dữ liệu bằng cách tiết kiệm thời gian và nguồn lực, và cung cấp việc quản lý tập trung cho các quản trị viên. Người dùng có thể được trao nhiều role và một role có thể được trao cho nhiều người sử dụng. Một role cũng có thể nằm trong một role khác và thừa hưởng đặc quyền.

3.3.3 Kiểm toán (Auditing)

Các cơ chế kiểm toán (*auditing mechanisms*): giám sát việc sử dụng tài nguyên hệ thống của người dùng. Các cơ chế này bao gồm hai giai đoạn:

- *Giai đoạn ghi vào nhật ký*: tất cả các câu hỏi truy nhập và câu trả lời liên quan đều được ghi lại (dù được trả lời hay bị từ chối).
- *Giai đoạn báo cáo*: các báo cáo của giai đoạn trước được kiểm tra, nhằm phát hiện các xâm phạm hoặc tấn công có thể xảy ra.

Các cơ chế kiểm toán thích hợp cho việc bảo vệ dữ liệu, bởi vì chúng hỗ trợ:

- Đánh giá phản ứng của hệ thống đối với một số dạng hiểm họa. Từ đó có thể phát hiện các điểm yếu và sự không đầy đủ của hệ thống.
- Phát hiện các xâm phạm chủ ý được thực hiện thông qua chuỗi các câu truy vấn.

Do quy mô của một môi trường cơ sở dữ liệu và tài nguyên cần thiết để hoàn thành một kiểm toán cơ sở dữ liệu là rất lớn, nên việc kiểm toán thường được thực hiện trong từng phần nhỏ, tập trung vào các chức năng cụ thể hoặc các khu vực tập trung. Kiểm toán các khu vực tập trung có thể bao gồm: kiểm toán bảo trì máy chủ, kiểm toán tài khoản quản trị, kiểm toán kiểm soát truy nhập, kiểm toán đặc quyền dữ liệu, kiểm toán mật khẩu, kiểm toán mã hóa, và kiểm toán hoạt động.

Bảo trì máy chủ

Việc bảo trì máy chủ nhằm đảm bảo các máy chủ hoạt động một cách hợp lý và chính xác. Kiểm toán bảo trì máy chủ bao gồm việc xem xét các phần mềm, công cụ cập nhật, chiến lược sao lưu, kiểm tra phiên bản ứng dụng, quản lý tài nguyên, và cập nhật phần cứng. Dưới đây là một danh sách các ví dụ về kiểm tra kiểm toán:

- Các bản vá lỗi bảo mật mới nhất được áp dụng
- Các cập nhật mới nhất của DBMS đã được áp dụng
- Phiên bản của DBMS được hỗ trợ
- Tồn tại một quy trình để duy trì các bản vá lỗi và phiên bản phần mềm
- ...

Tài khoản quản trị

Tài khoản quản trị là một phần quan trọng đối với an toàn cơ sở dữ liệu. Việc tài khoản người dùng được xử lý như thế nào là rất quan trọng để quản lý truy nhập và điều khiển đặc quyền. Kiểm toán tài khoản quản trị bao gồm: đánh giá cách thức mà người quản trị xác định và tạo ra các tài khoản người dùng, loại bỏ tài khoản người dùng, áp dụng chính sách bảo mật và phân nhóm, vai trò và đặc quyền. Một số kiểm tra kiểm toán mẫu bao gồm:

- Vai trò của người quản trị được xác định rõ ràng
- Tài khoản quản trị được phân bổ hợp lý
- Không được sử dụng tài khoản chung
- Phải khóa hoặc gỡ bỏ các tài khoản mặc định
- Kiểm tra tính toàn vẹn của việc sao lưu
- ...

Kiểm soát truy nhập

Kiểm soát truy nhập là hành động kiểm soát, xử lý, cho phép và phát hiện người dùng truy nhập vào cơ sở dữ liệu và tài nguyên của hệ thống. Kiểm soát

truy nhập là việc cần thiết để đảm bảo tính bí mật, tính toàn vẹn và tính sẵn sàng của hệ quản trị cơ sở dữ liệu. Việc kiểm toán kiểm soát truy nhập là công việc rất tốn thời gian và có thể yêu cầu việc đăng nhập vào cơ sở dữ liệu trong một khoảng thời gian nhất định. Một số kiểm tra kiểm toán mẫu bao gồm:

- Xác định chỉ địa chỉ IP đáng tin cậy có thể truy nhập cơ sở dữ liệu
- Dữ liệu nhạy cảm được truy nhập chỉ bởi những đối tượng hợp pháp
- Liên kết cơ sở dữ liệu có phù hợp không
- Những cơ sở dữ liệu riêng phải có kiểm soát truy nhập thích hợp
- Chỉ cho phép người dùng là người quản trị được phép truy nhập sao lưu và phục hồi thảm họa trong cơ sở dữ liệu.
- ...

Đặc quyền cơ sở dữ liệu

Việc giám sát đặc quyền người dùng chặt chẽ sẽ góp phần đảm bảo an toàn cơ sở dữ liệu. Đảm bảo tính phù hợp của đặc quyền trong một cuộc kiểm toán là công việc tốn nhiều thời gian nhất mà thường đòi hỏi khá nhiều sự hợp tác của các nhà quản trị mạng. Một số kiểm tra kiểm toán mẫu bao gồm:

- Xem xét cẩn thận các đặc quyền được cấp ngầm định
- Sử dụng nguyên tắc đặc quyền tối thiểu
- Hạn chế các đặc quyền cho thủ tục lưu trữ
- ...

Mật khẩu

Mật khẩu mạnh là rất quan trọng trong một môi trường an toàn, đó là hàng rào đầu tiên của hệ thống phòng thủ đối với những kẻ xâm nhập. Trong hầu hết các hệ thống quản lý cơ sở dữ liệu, việc xác thực người dùng được cấu hình bằng mật khẩu để đảm bảo an toàn. Kiểm toán quản lý mật khẩu liên quan đến việc xem xét lại các chính sách bằng văn bản, cấu hình máy chủ, và các tài khoản người dùng mặc định. Dưới đây là một số kiểm tra kiểm toán mẫu:

- Khả năng quản lý mật khẩu được kích hoạt trong các DBMS
- Mật khẩu mặc định đã được thay đổi hay chưa
- Mật khẩu không nên được lưu trữ trong cơ sở dữ liệu (nếu có thể)
- Nếu mật khẩu được lưu trữ trong cơ sở dữ liệu thì cần được mã hóa mạnh.
- ...

Mã hóa

Đối với việc mã hóa, một số kiểm tra kiểm toán mẫu bao gồm:

- Dữ liệu lưu trữ được mã hóa bằng kỹ thuật mã hóa mạnh
- Mã hóa được cấu hình chính xác
- Khóa đối xứng được sử dụng để mã hóa dữ liệu
- Dữ liệu nhạy cảm được ghi chép và gắn nhãn
- Mật khẩu được mã hóa trong khi đăng nhập từ xa vào cơ sở dữ liệu
- ...

Hoạt động

Kiểm toán hoạt động trong kiểm toán an toàn là một kỹ thuật thực hành tốt nhằm bảo vệ cơ sở dữ liệu an toàn. Nhiều thông tin có thể được phát hiện bằng việc sử dụng công cụ giám sát và các bản ghi nhật ký sự kiện. Một số kiểm tra kiểm toán mẫu bao gồm:

- Theo dõi đăng nhập không thành công
- Theo dõi truy vấn thất bại
- Theo dõi thay đổi đối với các siêu dữ liệu
- ...

Báo cáo kiểm toán bảo mật cơ sở dữ liệu

Bước cuối cùng của quá trình kiểm toán an toàn là một cuộc họp, trong đó kiểm toán viên hoặc ủy ban kiểm toán giao tiếp bằng lời nói và bằng văn bản để trình bày kết quả kiểm toán. Bản báo cáo kiểm toán sẽ cung cấp một cái nhìn chi tiết về kiểm toán nội bộ của tổ chức, bao gồm cả các lỗ hổng

bảo mật và các rủi ro, đồng thời trong một số trường hợp, đưa ra được những điểm mạnh của hệ thống.

Thông thường báo cáo kiểm toán bảo mật bao gồm các thành phần sau: các thông tin kiểm toán cơ bản, phạm vi kiểm toán, đối tượng kiểm toán, các phát hiện quan trọng, phương thức sử dụng để kiểm toán rủi ro và cuối cùng là đề xuất khắc phục.

3.4 CÁC CƠ CHẾ AN TOÀN TRONG HỆ QUẢN TRỊ ORACLE

Oracle là một hệ quản trị mạnh hiện nay với rất nhiều ưu điểm so với các hệ quản trị khác. Oracle không chỉ nhắm tới những doanh nghiệp lớn mà còn nhắm tới những doanh nghiệp trung bình và nhỏ. Cụ thể là Oracle Server có đủ các phiên bản thương mại từ Personal, Standard đến Enterprise (ngoài ra còn có Oracle lite).

Về phía các doanh nghiệp, Oracle có tính bảo mật cao, tính an toàn dữ liệu cao, dễ dàng bảo trì nâng cấp, cơ chế quyền hạn rõ ràng, ổn định với chi phí hợp lý.

Về phía những nhà phát triển, Oracle cũng tỏ ra rất có ưu điểm như dễ cài đặt, dễ triển khai và dễ nâng cấp lên phiên bản mới. Hơn nữa Oracle còn tích hợp thêm PL/SQL, là một ngôn ngữ lập trình có cấu trúc, tạo thuận lợi cho các lập trình viên viết các Trigger, Store Procedure, Package. Đây là điểm rất mạnh của Oracle so với các hệ quản trị hiện có trên thị trường.

Oracle, ngoài các kiểu dữ liệu thông thường còn có các kiểu dữ liệu đặc biệt khác góp phần mang lại sức mạnh cho Oracle như Blob, clob, Bfile,... Ngoài ra, có thể triển khai Oracle trên nhiều OS khác nhau (Windows, Solaris, Linux,...) mà không cần phải viết lại PL/SQL code. Có thể nhập vào (import) một dumpFile (backupFile) từ một máy chạy OS này sang OS khác hoặc từ một phiên bản thấp lên một phiên bản cao hơn mà không gặp bất cứ trở ngại nào.

Với những ưu điểm đó, trong phần này, chúng ta sẽ tìm hiểu một số cơ chế an toàn mạnh trong Oracle như: Cơ sở dữ liệu riêng ảo (VPD), an toàn

dựa vào nhẫn (OLS), kiểm toán mịn, An toàn nâng cao (Oracle Advanced Security), mã hóa, backup,...

3.4.1 Cơ sở dữ liệu riêng ảo (VPD)

3.4.1.1 Tổng quan về VPD

Khái niệm chung về VPD

Cơ sở dữ liệu riêng ảo (hay VPD – Virtual Private Database) trong Oracle còn được gọi là kiểm soát truy nhập mức mịn hay cơ chế an toàn mức hàng. VPD cung cấp tính năng bảo mật mức hàng cho cơ sở dữ liệu. Sau khi phát hành cùng với phiên bản Oracle 8i, VPD được hoan nghênh rộng rãi và ứng dụng trong nhiều lĩnh vực, từ giáo dục đến các dịch vụ tài chính và các ứng dụng phần mềm khác.

VPD cung cấp giải pháp bảo mật tối mức mịn trực tiếp trên các table, view, synonym. Nó gán trực tiếp các chính sách bảo mật lên các đối tượng cơ sở dữ liệu, và các chính sách sẽ tự động được thực hiện mỗi khi có một người dùng truy nhập dữ liệu đến các đối tượng đó.

Các ưu điểm của VPD

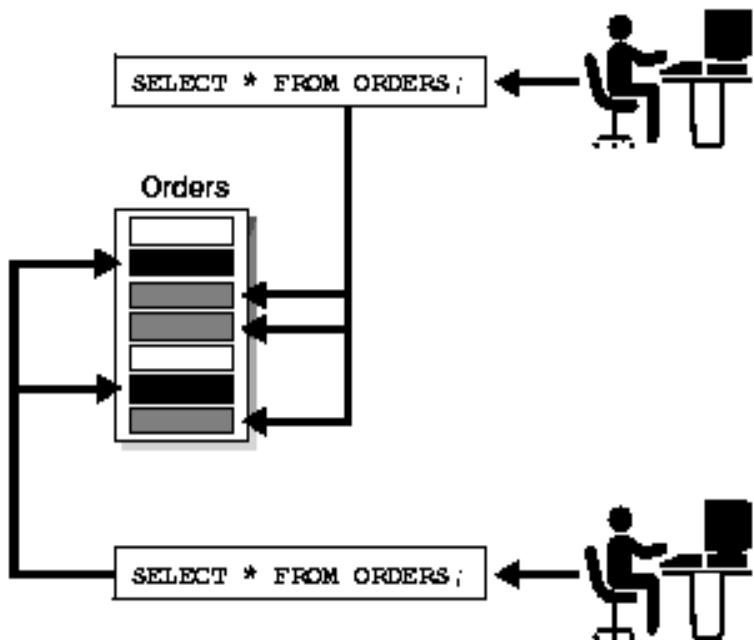
VPD mang lại rất nhiều lợi ích trong việc bảo mật cơ sở dữ liệu trong hệ quản trị cơ sở dữ liệu Oracle. Và dưới đây là một số lợi ích của VPD mà ta có thể nhận thấy dễ dàng đó là:

Chi phí thấp: các tổ chức có thể tiết kiệm được một khoản chi phí rất lớn bằng cách triển khai VPD trên toàn bộ cơ sở dữ liệu, thay vì phải thực thi các chính sách bảo mật giống nhau cho mỗi ứng dụng khi có truy nhập dữ liệu.

Trong suốt với người dùng: Người dùng có thể truy nhập dữ liệu thông qua một câu truy vấn hay một ứng dụng mà không hề biết tới các chính sách bảo mật được áp dụng như thế nào. Bởi các chính sách bảo mật được đính kèm với cơ sở dữ liệu và được thực thi một cách tự động trên máy chủ cơ sở dữ liệu.

Cơ hội kinh doanh: Trước đây, các công ty không thể cho khách hàng, đối tác truy nhập trực tiếp vào hệ thống cơ sở dữ liệu của họ bởi họ chưa có cách

nào để làm an toàn dữ liệu của họ. Các công ty mẹ thì không thể có được tất cả dữ liệu của các công ty con trên máy chủ cơ sở dữ liệu, bởi các công ty con sẽ có những dữ liệu nhạy cảm không thể tiết lộ. Hiện nay, tất cả những khó khăn ấy đã được giải quyết, bằng cách kiểm soát truy nhập mức mịn, dữ liệu sẽ được bảo mật trên máy chủ, và việc truy nhập được giới hạn thông qua chính sách VPD.



Hình 3.15. Cơ sở dữ liệu riêng ảo

3.4.1.2 Các thành phần của chính sách VPD

Ngữ cảnh ứng dụng:

Ngữ cảnh ứng dụng là một tập các cặp tên – giá trị, được lưu trong bộ nhớ, nó được xác định, thiết lập và lấy ra bởi người dùng và các ứng dụng. Các giá trị liên quan được nhóm lại thành một nhóm được truy nhập theo không gian tên miền (namespace) hay tên của nó.

Bằng cách lưu trữ các giá trị và các thuộc tính trong bộ nhớ, sau đó chia sẻ chúng dựa trên ngữ cảnh, sẽ giúp việc truy xuất các giá trị nhanh chóng hơn

Hàm PL/SQL

Hàm PL/SQL tổ chức các lệnh theo từng khối lệnh. Một khối lệnh PL/SQL cũng có thể có các khối lệnh con khác ở trong nó.

Cấu trúc đầy đủ của một khối lệnh PL/SQL bao gồm:

DECLARE / Phần khai báo - Không bắt buộc */*

Khai báo các biến sử dụng trong phần thân;

BEGIN / Phần thân */*

Đoạn lệnh thực hiện;

EXCEPTION / Phần xử lý lỗi - Không bắt buộc */*

Xử lý lỗi xảy ra;

END;

Các chức năng cần phải có:

- ✓ Nó phải là đối số của một tên lược đồ và một đối tượng đầu vào (table, view, synonyms).
- ✓ Các giá trị trả về của mệnh đề WHERE được tạo ra, có kiểu dữ liệu VARCHAR2.
- ✓ Mệnh đề WHERE phải được tạo ra chính xác, và giống nhau cho tất cả người dùng đăng nhập.

Chính sách an toàn

- ✓ Chính sách là một kỹ thuật quản lý hàm của VPD.
- ✓ Chính sách cho phép ta thêm vào việc kiểm soát truy nhập ở mức mịn, chẳng hạn như xác định các kiểu câu lệnh SQL mà chính sách ảnh hưởng tới.
- ✓ Khi người dùng truy nhập tới dữ liệu trong CSDL, thì chính sách tự động có hiệu lực.

3.4.1.3 Cấu hình một chính sách VPD

3.4.1.3.1 Giới thiệu về chính sách VPD

Sau khi ta tạo ra một hàm để định nghĩa các hành động, ta phải kết hợp hàm này với một bảng để chính sách VPD được thi hành. Chính sách chính là một kỹ thuật cho việc quản lý hàm VPD. Chính sách cũng cho phép ta thêm vào việc kiểm soát truy nhập ở mức mịn, chẳng hạn như xác định các kiểu câu lệnh SQL hoặc các cột cụ thể mà chính sách ảnh hưởng tới. Khi mà người dùng có truy nhập tới dữ liệu trong đối tượng cơ sở dữ liệu này, thì chính sách tự động có hiệu lực. Để quản lý chính sách VPD ta sử dụng gói DBMS_RLS.

3.4.1.3.2 Gắn một chính sách vào một bảng CSDL

Để gắn một chính sách lên table, view, synonym, ta sử dụng thủ tục DBMS_RLS. ADD_POLICY. Ta phải chỉ rõ table, view, synonym được áp dụng chính sách và tên chính sách đó. Ta cũng có thể chỉ ra thông tin khác chẳng hạn như kiểu của câu lệnh (statements) mà chính sách điều khiển (SELECT, INSERT, UPDATE, DELETE, CREATE INDEX, or ALTER INDEX).

Ví dụ: gắn một chính sách VPD có tên là secure_update lên bảng HR. EMPLOYEES, hàm gắn chính sách là check_updates

BEGIN

DBMS_RLS.ADD_POLICY (

*objectSchema => 'hr',
object_name => 'employees',
policy_name => 'secure_update',
policy_function => 'check_updates');*

END;

Nếu hàm được tạo ra bên trong gói (package) thì nó bao gồm cả tên gói:

policy_function => 'pkg. check_updates',

3.4.1.3.3 Thực thi các chính sách trên mệnh đề cụ thể kiểu SQL

Ta có thể thi hành các chính sách VPD cho các lệnh SELECT, INSERT, UPDATE, DELETE. Nếu không chỉ rõ lệnh nào thì mặc định Oracle chỉ định lệnh SELECT, INSERT, UPDATE, DELETE, ko có INDEX. Bất kì sự kết hợp câu lệnh nào đều sử dụng tham số statement_types trong thủ tục DBMS_RLS.ADD_POLICY, kèm theo là danh sách lệnh được đặt trong dấu ngoặc đơn.

Ví dụ: Chỉ định lệnh SELECT và INDEX cho chính sách sau

```
BEGIN  
  
    DBMS_RLS.ADD_POLICY(  
  
        object_schema      => 'hr',  
        object_name        => 'employees',  
        policy_name        => 'securejupdate',  
        policy_function    => 'check_updates',  
        statement_types     => 'SELECT,INDEX');  
  
END;
```

3.4.2 An toàn dựa vào nhãn trong Oracle

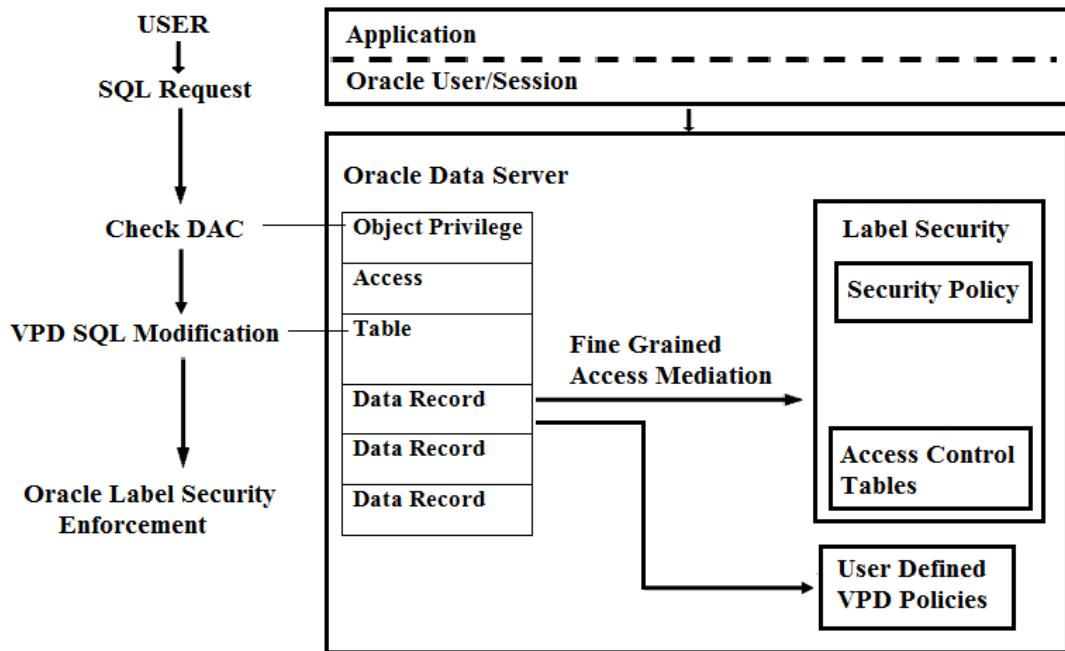
3.4.2.1 Tổng quan về OLS

3.4.2.1.1 Kiến trúc OLS

OLS được xây dựng trên công nghệ VPD được chuyển giao trong các phiên bản hệ quản trị cơ sở dữ liệu Oracle Enterprise. OLS cho phép bảo vệ dữ liệu của các bảng đến mức hàng - mức bản ghi (row level).

OLS cho phép định nghĩa một chính sách an toàn (security policy) được thực thi bằng cách gắn cho các bản ghi trong bảng bởi các nhãn an toàn. Nhãn này cũng thể hiện quyền mà một người dùng cần có để có thể đọc hay ghi dữ liệu trong các bản ghi. Khi áp dụng chính sách an toàn vào một bảng, bảng đó

sẽ được bổ sung thêm một cột để chứa các nhãn an toàn gắn với từng bản ghi của nó.



Hình 3.16 Kiến trúc OLS

3.4.2.1.2 Các tính năng của OLS

OLS cung cấp 4 khía cạnh của kiểm soát truy nhập dựa vào nhãn:

- ✓ Nhãn người dùng cung cấp thông tin về quyền hạn của người dùng.
- ✓ Nhãn dữ liệu cho thấy độ nhạy cảm của thông tin trong hàng đó.
- ✓ Chính sách đặc quyền của người dùng có thể cho phép bỏ qua một số khía cạnh của kiểm soát truy nhập dựa vào nhãn.
- ✓ Tùy chọn thực thi chính sách của một bảng xác định các khía cạnh khác nhau về cách điều khiển truy nhập thực thi để đọc, ghi lên bảng đó.

3.4.2.2 Nhãn dữ liệu và nhãn người dùng

3.4.2.2.1 Nhãn dữ liệu và nhãn người dùng

Nhãn dữ liệu cho thấy mức độ và tính chất nhạy cảm của dữ liệu trong hàng và quy định các tiêu chí bổ sung mà một người dùng phải đáp ứng để được truy nhập vào hàng đó.

Nhãn người dùng thể hiện mức độ nhạy cảm của người dùng cộng với các compartment và các group để hạn chế truy nhập của người dùng trên dữ liệu được gắn nhãn. Mỗi người dùng được gắn một level, các compartment, và các group, và mỗi phiên có thể hoạt động trong phạm vi truy nhập dữ liệu đó.

3.4.2.2.2 Các thành phần của nhãn

Level: là thành phần bắt buộc, phân cấp, thể hiện mức nhạy cảm của thông tin

Compartment: Là thành phần tùy chọn và không phân cấp. Nó là các danh mục liên quan đến các dữ liệu được dán nhãn. Ví dụ, compartment là các phòng ban làm việc của một tổ chức.

Group: Là thành phần tùy chọn và phân cấp, định danh cho tổ chức sở hữu hoặc truy nhập vào dữ liệu. Group có ích trong việc phân cấp người dùng. Ví dụ, Group có thể là các chi nhánh của một công ty, các vùng miền của một đất nước.

3.4.2.2.3 Cú pháp nhãn và các kiểu nhãn

Sau khi xác định được các compartment của nhãn, người quản trị tạo ra nhãn dữ liệu bằng cách kết hợp ba phần là: level, compartment, và group bằng cách hoán vị các compartment, người quản trị có thể xác định được nhãn dữ liệu hợp lệ trong cơ sở dữ liệu.

Người quản trị có thể chọn giao diện đồ họa Oracle Enterprise hoặc dùng dòng lệnh để tạo các nhãn. Tạo chuỗi ký tự của nhãn dùng cú pháp sau:

*LEVEL:COMPARTMENT1,...,COMPARTMENTn:GROUP1,...
,GROUPn*

Chuỗi văn bản quy định các nhãn có thể có tối đa 4.000 ký tự, bao gồm cả ký tự chữ, số, khoảng trắng, gạch dưới. Có thể nhập các nhãn bằng chữ hoa, chữ thường, hoặc kết hợp, nhưng chuỗi được lưu trữ trong từ điển dữ liệu là dạng chữ hoa. Dấu phẩy (,) được dùng như các dấu phân cách giữa các compartment hoặc các group.

Ví dụ, người quản trị có thể tạo ra các nhãn hợp lệ như thế này:

SENSITIVE:FINANCIAL,CHEMICAL:EA_REGION,WE_REGION

SENSITIVE::WE_REGION

Khi một nhãn dữ liệu hợp lệ được tạo ra, có hai vấn đề xảy ra:

- Nhãn tự động được chỉ định như là một nhãn dữ liệu hợp lệ. OLS cũng có thể tạo ra các nhãn dữ liệu hợp lệ một cách tự động trong thời gian chạy, từ những người dùng được xác định trước trong Oracle Internet Directory.

- Một thẻ số được kết hợp với đoạn văn bản đại diện cho nhãn. Nó là số chứ không phải là chuỗi văn bản, và được lưu trong cột chính sách nhãn của bảng được bảo vệ.

3.4.2.2.4 Quản lý nhãn

Một người dùng có thể truy nhập dữ liệu chỉ trong phạm vi nhãn nhạy cảm của người dùng đó.

OLS cung cấp giao diện quản trị để xác định và quản lý các nhãn dùng trong một cơ sở dữ liệu. Để định nghĩa nhãn trong Oracle, ta có thể dùng gói OLS hoặc dùng Oracle Enterprise Manager. Ban đầu, người quản trị phải xác định level, compartment, và group để tạo các nhãn cho người dùng và thiết lập các nhãn dữ liệu hợp lệ dựa vào nội dung của bảng cơ sở dữ liệu.

Người quản trị có thể áp dụng một chính sách tới bảng cá nhân trong cơ sở dữ liệu hoặc toàn bộ lược đồ ứng dụng. Cuối cùng, người quản trị chỉ định cho mỗi người dùng cơ sở dữ liệu các thành phần nhãn (và ưu tiên, nếu cần thiết) cho chức năng công việc của người dùng.

3.4.2.3 Cấu hình một chính sách OLS

Cần có 5 bước để thực thi OLS trong Oracle:

- **Bước 1** (Tạo chính sách OLS): Chính sách này bao gồm các nhãn (label), quyền người dùng (user authorization), đối tượng cơ sở dữ liệu cần bảo vệ (protected database object).

- **Bước 2** (Định nghĩa các thành phần nhãn - label component): mỗi một label bao gồm 3 thành phần: level, compartment, group.

- **Bước 3** (Tạo các label để sử dụng): Từ các label component đã định nghĩa ở bước 2, tạo một tập các nhãn dựa vào chính sách an toàn phù hợp với ứng dụng.

- **Bước 4** (Áp dụng chính sách an toàn trên cho các bảng hay các lược đồ): Sau bước này, các bảng hay lược đồ đó sẽ được tạo thêm một cột chứa nhãn an toàn của từng hàng.

- **Bước 5** (Gán nhãn cho các user hay các ứng dụng): Bước này thực hiện gán nhãn cho người dùng một cách phù hợp với từng người dùng. Sau đó OLS sẽ so sánh nhãn của một người dùng với nhãn dữ liệu gán cho từng hàng để đưa ra quyết định truy nhập. Một người dùng chưa được gán nhãn sẽ không thể truy nhập vào bảng đó.

3.4.2.4 Sử dụng VPD để thi hành chính sách OLS

Ta có thể sử dụng chính sách VPD để kiểm soát cột hoặc kiểm soát truy nhập mức hàng dựa trên ủy quyền chính sách OLS cho người dùng. Nói chung, ta phải thực hiện các bước sau:

- **Bước 1:** Khi ta tạo các chính sách OLS, không áp dụng chính sách lên bảng mà ta muốn bảo vệ. (Chính sách VPD mà ta tạo ra sẽ thực hiện việc này). Trong thủ tục SA_SYSDBA.CREATE_POLICY, thiết lập các tham số default_options để NO_CONTROL.

- **Bước 2:** Tạo ra các nhãn OLS và ủy quyền cho người sử dụng như bình thường.

- **Bước 3:** Khi ta tạo các chính sách VPD, làm như sau:

Sử dụng hàm PL/SQL tạo chức năng cho chính sách, sử dụng OLS quản lý chức năng để so sánh nhãn người dùng với các nhãn dữ liệu mà ta đã tạo ở bước 2. Nếu nhãn người dùng có độ nhạy cảm cao hơn nhãn dữ liệu thì người dùng được truy nhập vào cột. Nếu nhãn dữ liệu có độ nhạy cảm cao hơn thì người dùng bị từ chối.

Trong định nghĩa VPD, áp dụng chức năng vừa tạo lên các bảng mà ta muốn bảo vệ. Trong thủ tục DBMS_RLS.ADD_POLICY, sử dụng tham số sec_relevant_cols (cột mức nhạy cảm) và tham số sec_relevant_cols_opt

trong chức năng để hiển thị hoặc ẩn các cột dựa trên ủy quyền OLS của người dùng.

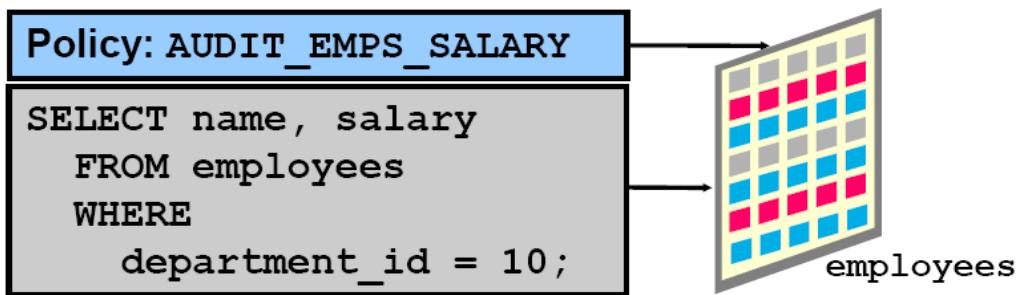
3.4.3 Cơ chế Kiểm toán mịn (Fine-grained auditing)

Kiểm toán mức mịn cho phép giám sát và ghi lại việc truy nhập dữ liệu dựa trên nội dung của dữ liệu. Với kiểm toán mức mịn, ta có thể định nghĩa một chính sách kiểm toán trên một bảng và các cột tùy chọn. Oracle sử dụng gói thủ tục PL/SQL DBMS_FGA để cấu hình và quản lý Kiểm toán mức mịn.

Kiểm toán mức mịn cung cấp cơ chế điều khiển tốt hơn và mức chi tiết nhỏ hơn so với các phương pháp kiểm toán thông thường như: kiểm toán câu lệnh, kiểm toán đặc quyền, kiểm toán đối tượng lược đồ. Ta biết rằng, trigger chỉ được viết trên câu lệnh DML như: insert, update, delete mà không phải trên câu lệnh select. Do đó, nếu muốn bắt sự kiện bởi câu lệnh select trên một bảng hoặc trên một cột cụ thể của bảng thì ta phải sử dụng Kiểm toán mức mịn.

Trong Oracle 9i, kiểm toán mức mịn chỉ được hỗ trợ câu lệnh select, nhưng trong Oracle 10g, kiểm toán mức mịn hỗ trợ tất cả các câu lệnh DML. Vì vậy, tất cả các câu lệnh insert, update, delete, select nó đều được bắt bằng cách sử dụng kiểm toán mức mịn và có thể được xem qua từ điển dữ liệu DBA_FGA_AUDIT_TRAIL.

Trong Oracle 10g, Kiểm toán mức mịn thực hiện kiểm toán thông qua một gói tên là DBMS_FGA. Gói này cho phép thực hiện kiểm toán tại một mức chi tiết thấp với bất kỳ bảng nào trong cơ sở dữ liệu, dựa vào một đối tượng đặc biệt được gọi là chính sách kiểm toán mức mịn. Gói DBMS_FGA có nhiều chương trình con, cho phép ta thiết lập các điều kiện kiểm toán và xác định các cột kiểm toán trong một bảng. Khi các cột thỏa mãn điều kiện kiểm toán thì Kiểm toán mức mịn tạo ra một bản ghi kiểm toán cho các câu truy vấn SQL liên quan đến các cột đó.



Hình 3.17: Ví dụ về chính sách kiểm toán

Bản ghi kiểm toán có các thông tin chi tiết như: người sở hữu (owner), tem thời gian (timestamp), loại câu lệnh, ... Tuy nhiên, nó không cung cấp thông tin về sự thay đổi đã xảy ra đối với dữ liệu. Các bản ghi kiểm toán rất hữu ích cho các DBA hoặc người dùng muốn phân tích các hoạt động diễn ra trên bảng cơ sở dữ liệu. Bảng dưới đây, chỉ ra các thủ tục trong gói DBMS_FGA:

ADD_POLICY Procedure	Tạo ra một chính sách kiểm toán bằng cách sử dụng các thuộc tính được cung cấp như điều kiện kiểm toán
DISABLE_POLICY Procedure	Vô hiệu hóa một chính sách kiểm toán
DROP_POLICY Procedure	Xoá một chính sách kiểm toán
ENABLE_POLICY Procedure	Cho phép một chính sách kiểm toán

Bảng 3.7: Các thủ tục của gói DBMS_FGA

Cú pháp câu lệnh SQL tạo thủ tục ADD_POLICY như sau:

```

DBMS_FGA.ADD_POLICY
(
    object_schema VARCHAR2,
    object_name VARCHAR2,

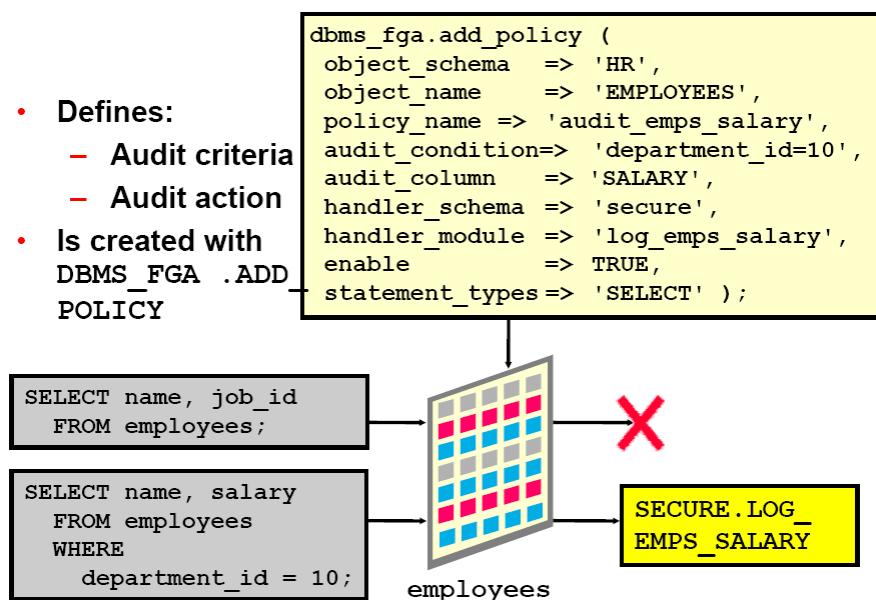
```

```

policy_name VARCHAR2,
audit_condition VARCHAR2,
audit_column VARCHAR2,
handler_schema VARCHAR2,
handler_module VARCHAR2,
enable BOOLEAN,
statement_types VARCHAR2,
audit_trail BINARY_INTEGER IN DEFAULT,
audit_column_opts BINARY_INTEGER IN DEFAULT
);

```

Hình vẽ dưới đây là một ví dụ về thủ tục tạo chính sách kiểm toán mịnh DBMS_FGA.ADD_POLICY.



Hình 3.18 Ví dụ về thủ tục DBMS_FGA.ADD_POLICY

3.4.4 Các cơ chế an toàn khác

Ngoài những cơ chế an toàn trên, Oracle còn cung cấp nhiều cơ chế an toàn khác mà người dùng có thể lựa chọn để phục vụ cho nhu cầu bảo mật cơ sở dữ liệu của mình như: Oracle Advanced Security, Oracle Secure Backup,...

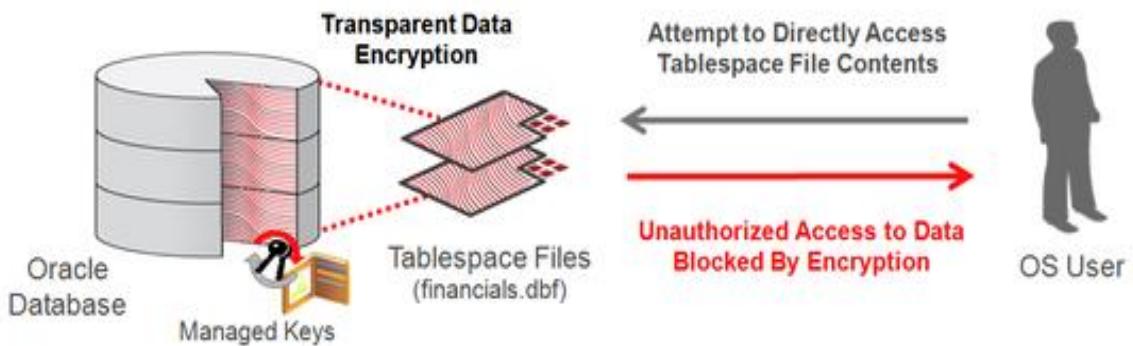
3.4.4.1 Oracle Advanced Security

Bảo vệ dữ liệu đòi hỏi một sự tiếp cận theo chiều sâu bao gồm cả vấn đề dự phòng, theo dõi, và quản lý hành chính. Oracle Advanced Security là cơ chế an toàn nâng cao trong Oracle, cho phép kiểm soát phòng ngừa, giúp giải quyết nhiều yêu cầu đặt ra, ngăn chặn các hành vi vi phạm dữ liệu, và bảo vệ thông tin. Ví dụ, dữ liệu thẻ tín dụng có thể được tự động mã hóa khi lưu trữ và cùng một lúc, giải mã trước khi rời khỏi cơ sở dữ liệu trong kết quả truy vấn. Hai khả năng này rất quan trọng trong việc thực hiện các quy định bảo mật và Chuẩn bảo mật dữ liệu thẻ thanh toán trong công nghiệp (PCI-DSS).

Oracle Advanced Security cung cấp tính riêng tư, tính toàn vẹn, xác thực, cấp quyền truy nhập với nhiều cách thức khác nhau. Chẳng hạn, ta có thể cấu hình thành phần Oracle Net native encryption hoặc cách khác là sử dụng SSL để đảm bảo tính riêng tư. Oracle Advanced Security cũng cung cấp nhiều phương pháp xác thực mạnh, bao gồm: Kerberos, smart cards, và các chứng chỉ số. Oracle Advanced Security cung cấp hai tính năng an toàn sau:

- Mã hóa dữ liệu trong suốt
- Xác thực mạnh.

Mã hóa dữ liệu trong suốt (Transparent Data Encryption - TDE) là biện pháp bảo vệ dữ liệu nhạy cảm chống lại truy nhập trái phép từ bên ngoài của môi trường cơ sở dữ liệu bằng cách mã hóa dữ liệu. Nó ngăn cản người dùng hệ điều hành trực tiếp truy nhập vào dữ liệu nhạy cảm bằng cách kiểm tra các nội dung của tập tin cơ sở dữ liệu. Mã hóa và biện tập dữ liệu trong suốt giúp ngăn chặn truy nhập trái phép vào các thông tin nhạy cảm ở lớp ứng dụng, trong hệ điều hành, trên phương tiện sao lưu, và trong cơ sở dữ liệu trích xuất.



Hình 3.19. Mã hóa dữ liệu trong suốt trong Oracle

TDE trong suốt với những ứng dụng bởi vì dữ liệu được mã hóa tự động khi ghi để lưu trữ và được tự động giải mã khi đọc. Các cơ chế kiểm soát truy nhập được thực thi tại các cơ sở dữ liệu và lớp ứng dụng vẫn có hiệu lực. Hơn nữa, truy vấn SQL được giữ nguyên, và không yêu cầu thay đổi mã nguồn ứng dụng hay cấu hình.

Quá trình mã hóa và giải mã rất nhanh vì TDE thúc đẩy Oracle tối ưu hóa bộ nhớ đệm. Ngoài ra, TDE sử dụng công nghệ tăng tốc phần cứng CPU dựa trên nền tảng Intel® AES-NI và nền tảng Oracle SPARC T-Series, bao gồm cả Oracle Exadata và SPARC SuperCluster. TDE có lợi ích hơn nữa từ công nghệ Exadata Smart Scans, giải mã dữ liệu song song trên nhiều đơn vị lưu trữ, và công nghệ Exadata Hybrid Columnar Compression cho phép giảm tổng số các hoạt động mã hóa cần thực hiện. Do đó TDE mã hóa và giải mã dữ liệu rất nhanh.

TDE cung cấp một kiến trúc quản lý khóa mã hai tầng bao gồm: các khóa mã hóa dữ liệu và khóa mã hóa chính. Khóa chính được lưu trữ bên ngoài của cơ sở dữ liệu trong một Oracle Wallet (ví). Chức năng quản lý khóa đi kèm, cung cấp khả năng thay đổi khóa mà không cần mã hóa lại tất cả dữ liệu và cung cấp cơ chế quản lý khóa trong toàn bộ vòng đời của nó.

TDE có thể được triển khai dễ dàng. Nó được cài đặt mặc định như một phần của quá trình cài đặt cơ sở dữ liệu. Dữ liệu có sẵn có thể được mã hóa

ngay trên hệ thống bởi Oracle Online Table hoặc có thể được mã hóa offline trong một chu trình được bảo trì nghiêm ngặt.

3.4.4.2 Oracle Secure Backup (OSB)

OSB cho phép bảo vệ dữ liệu đáng tin cậy thông qua hệ thống tập tin sao lưu. OSB hỗ trợ băng từ, ổ đĩa, môi trường SAN, Gigabit Ethernet, các môi trường SCSI bằng cách sử dụng định dạng tiêu chuẩn. OSB là một phần của giải pháp lưu trữ Oracle, nó làm giảm tính phức tạp và giảm chi phí cho việc mua phần mềm bổ sung. OSB cung cấp khả năng mở rộng phân phối và khôi phục sao lưu dự phòng. Nó làm giảm chi phí phần mềm bằng cách:

- Tích hợp với ngăn xếp Oracle để người dùng có thể dễ dàng sử dụng một giải pháp Oracle duy nhất nhằm bảo vệ dữ liệu từ đĩa đến băng từ.
- Sử dụng kỹ thuật duy nhất được hỗ trợ từ nhà cung cấp cho cơ sở dữ liệu và tập tin hệ thống sao lưu và khôi phục trên băng từ.

Ta có thể sử dụng OSB kết hợp với bộ quản lý phục hồi (Recovery Manager - RMAN) để sao lưu và khôi phục cơ sở dữ liệu và tập tin trên băng từ mà không cần dùng một phần mềm quản lý khác. OSB hỗ trợ IPv4, IPv6 và môi trường IPv4/IP6 hỗn hợp trên tất cả các nền tảng hỗ trợ IPv6.

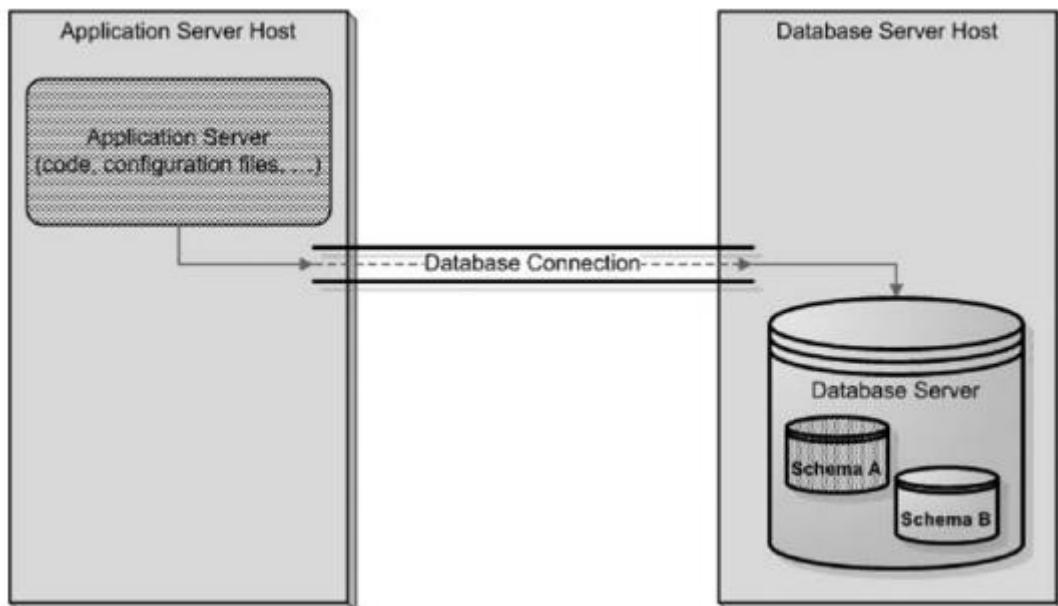
3.5 CÂU HỎI ÔN TẬP

1. Nêu kiến trúc của các hệ quản trị: Oracle, MySQL, SQL Server.
2. Nêu một vài lỗ hỏng phổ biến của Oracle, MySQL, SQL Server.
3. Quá trình xác thực nhằm mục đích gì? Phân biệt xác thực mức hệ điều hành, mức cơ sở dữ liệu và xác thực bên thứ ba.
4. Nêu một số cơ chế xác thực trong Oracle, MySQL, SQL Server.
5. Chính sách mật khẩu yêu cầu những vấn đề gì?
6. Mô tả yêu cầu cấp quyền và kiểm toán chung trong các hệ quản trị
7. Nêu một số cơ chế an toàn trong Oracle.

CHƯƠNG 4. AN TOÀN ỨNG DỤNG

4.1 GIỚI THIỆU

Trong những năm gần đây, lĩnh vực an toàn ứng dụng đã nhận được nhiều sự quan tâm, đặc biệt là các ứng dụng chạy trên internet hiện đang là mục tiêu cho nhiều cuộc tấn công, trong đó ứng dụng thường là phương tiện để kẻ tấn công có thể đạt được các mục tiêu tấn công vào dữ liệu ứng dụng được lưu trữ trong cơ sở dữ liệu. Do đó, mục đích của an toàn ứng dụng chính là đảm bảo an toàn cho dữ liệu của ứng dụng, bởi vì hầu hết các dữ liệu ứng dụng (các dữ liệu quan trọng như: dữ liệu tài chính, hồ sơ bệnh nhân, và thông tin khách hàng) được lưu trữ trong các cơ sở dữ liệu quan hệ. Việc đảm bảo an toàn cho dữ liệu của ứng dụng thực chất là đảm bảo an toàn cho việc truy nhập vào cơ sở dữ liệu. Hơn nữa, đối tượng truy cập chính vào dữ liệu là các ứng dụng, mà bản thân các ứng dụng lại chứa các lỗ hổng an toàn, đây chính là nguyên nhân gây ra các vấn đề cho an toàn của cơ sở dữ liệu.



Hình 4.1 Mô hình ứng dụng cơ sở dữ liệu

Hình 4.1 chỉ ra mô hình cơ bản của một ứng dụng cơ sở dữ liệu, trong đó cơ sở dữ liệu (hoặc một lược đồ cụ thể) là một phần của ứng dụng, do ứng dụng kiểm soát và quản lý. Theo quan điểm này, ứng dụng có quyền truy nhập đầy đủ đến các đối tượng trong lược đồ, do đó vấn đề an toàn nên được kiểm soát bằng tầng ứng dụng. Mô hình này cũng thể hiện rằng, ứng dụng là con đường thông suốt và duy nhất để truy nhập vào cơ sở dữ liệu, bất kỳ một sự cố an ninh nào xảy tại lớp ứng dụng, đều có thể dẫn đến các sự cố nghiêm trọng tại lớp dữ liệu. Chương này sẽ tìm hiểu các điểm yếu ở tầng ứng dụng và các biện pháp bảo vệ để hạn chế các tấn công vào cơ sở dữ liệu được xuất phát từ tầng ứng dụng.

4.2 VẤN ĐỀ QUẢN LÝ TÀI KHOẢN VÀ MẬT KHẨU NGƯỜI DÙNG

Tương tự như các mô hình an toàn khác, mô hình an toàn cơ sở dữ liệu dựa trên các tiến trình xác thực và phân quyền. Mỗi cơ sở dữ liệu duy trì danh sách các định danh của người dùng. Mỗi khi người dùng truy nhập vào cơ sở dữ liệu, tên và mật khẩu trong danh sách định danh sẽ được sử dụng để xác thực. Sau khi quá trình xác thực đã thành công, cơ sở dữ liệu sẽ xác định tập các quyền mà người dùng được phép thực hiện trên cơ sở dữ liệu.

Vì vậy, một điều hiển nhiên là để đáp ứng được yêu cầu mô hình an toàn đặt ra, tên và người dùng phải được quản lý một cách an toàn. Nếu điều này không được thực hiện tốt thì toàn bộ mô hình sẽ trở nên vô nghĩa. Ví dụ: nếu một người nào đó trong tổ chức có thể truy nhập vào một trang web nội bộ và họ có thể tìm được mật khẩu của một người quản trị cơ sở dữ liệu thì rõ ràng các biện pháp bảo vệ cơ sở dữ liệu không còn giá trị. Một ví dụ khác, một thống kê đã chỉ ra rằng khoảng 50% các ứng dụng viết bằng Java lưu trữ tên và mật khẩu cơ sở dữ liệu dưới dạng rõ trong tệp cấu hình của máy chủ ứng dụng, vì vậy mật khẩu cơ sở dữ liệu dễ dàng bị đánh cắp, dẫn đến mất an toàn cho cơ sở dữ liệu của hệ thống.

4.2.1 Lỗ hổng trong quản lý mật khẩu cơ sở dữ liệu

Lỗ hổng này đề cập đến vấn đề tên và mật khẩu của cơ sở dữ liệu được sử dụng và lưu trữ không an toàn. Có nhiều nguyên nhân dẫn đến lỗ hổng này, trong đó một nguyên nhân chính là tên và mật khẩu được lưu trữ dưới dạng rõ trong tệp cấu hình. Dẫn đến kẻ tấn công có thể lợi dụng để làm hại đến các tài nguyên của máy chủ ứng dụng, hoặc có thể đạt được việc truy nhập vào cơ sở dữ liệu một cách hợp pháp. Đây là một lỗ hổng bảo mật nghiêm trọng vì định danh được sử dụng bằng ứng dụng để truy nhập vào trong cơ sở dữ liệu thường có toàn quyền đối với lược đồ cơ sở dữ liệu, thậm chí là toàn quyền với cơ sở dữ liệu.

Mặc dù môi trường ứng dụng có thể khác nhau, nhưng lỗ hổng này không gắn liền với các công cụ ứng dụng cụ thể, nó chỉ đơn giản là một hệ quả của sự lựa chọn cấu hình không hợp lý. Hầu hết lỗ hổng này được tạo ra là do sự bất cẩn của người phát triển ứng dụng.

Dưới đây là một số ví dụ về lỗ hổng bảo mật có trong các ứng dụng Java trong các thủ tục kết nối vào cơ sở dữ liệu. Các thủ tục này được quản lý bằng máy chủ ứng dụng Java. Khi một ứng dụng cần truy nhập vào cơ sở dữ liệu, nó gọi đến một thủ tục kết nối, kết quả của thủ tục này sẽ trả về một nguồn kết nối dữ liệu phục vụ cho các hoạt động của ứng dụng.

Nguồn dữ liệu này được xem như một phần của hạ tầng máy chủ và được quản lý bởi máy chủ để cung cấp tài nguyên cho người phát triển ứng dụng. Trong phiên bản cũ của máy chủ ứng dụng JAVA, nguồn dữ liệu là một phần của máy chủ ứng dụng (Ví dụ: Webphere của IBM hoặc Weblogic của BEA) được thực hiện bên trong các thư viện độc quyền và là một phần của các máy chủ này. Trong các phiên bản mới hơn của JAVA, các thư viện JDBC sẽ cung cấp nguồn dữ liệu. Trong cả hai trường hợp trên, công việc này đều được làm bằng máy chủ ứng dụng, việc cấu hình cho nguồn này được thực hiện dựa trên các công cụ quản trị và các tệp cấu hình mà nó tạo ra hạ tầng máy chủ.

Mật khẩu được lưu dưới dạng rõ có thể do bất cẩn hoặc có thể là kết quả của sai sót ở lớp ứng dụng. Ví dụ WebLogic của BEA, máy chủ và

các phiên bản WebLogic Express 6.1 service pack6, phiên bản 7.0 service pack 4, và các phiên bản 8.1 service pack2 đều có một lỗ hổng có thể gây ra các mật khẩu cơ sở dữ liệu bị phơi bày dưới dạng rõ trong tệp config.xml. Hơn nữa, định nghĩa kết nối có thể được đặt trong một tập tin cấu hình văn bản rõ như sau:

```
weblogic.jdbc.connectionPool.eng=\  
    url=jdbc:weblogic:oracle,\  
    driver=weblogic.jdbc.oci.Driver,\  
    loginDelaySecs=2,\  
    initialCapacity=50,\  
    capacityIncrement=10,\  
    maxCapacity=100,\
```

Đoạn cấu hình này cho thấy có một nguồn dữ liệu bao gồm 50 kết nối ban đầu bằng cách sử dụng tên người dùng là *scott* và mật khẩu là *tiger* cho máy chủ *Oracle* có tên dịch vụ là *ORCL*.

Như đã thấy, bất kỳ kẻ tấn công hoặc người dùng nào truy nhập vào tập tin trên đều có thể bắt đầu truy nhập vào cơ sở dữ liệu bằng cách sử dụng tài khoản này trong đoạn mã. Để khắc phục hạn chế này cần tải về một bản vá cho lỗ hổng WebLogic (tìm hiểu thêm tại địa chỉ sau: http://ev2dev.bea.com/resource/library/advisoriesnotifications/BEA04_53.00.jsp)

Môi trường khác nhau có các định dạng khác nhau, tuy nhiên về cơ bản chúng có lỗ hổng và kịch bản tương tự. Ví dụ thứ hai, nguồn dữ liệu tài nguyên đăng ký với một máy chủ ứng dụng java thường được sử dụng để xác định một kết nối đến một cơ sở dữ liệu. Một nguồn dữ liệu định nghĩa trong máy chủ ứng dụng iPlanet của Sun có thể xác định được bằng cách sử dụng đoạn mã XML sau đây:

```
<ias-resources>
<resource>
<jndi-name>jdbc/ORCL</jndi-name>
<jdbc>
<database>ORCL</database>
<datasource>ORCL</datasource>
<username>scott</username>
<password>tiger</password>
<driver-type>ORACLE_OCI</driver-type>
</jdbc>
</resource>
</ias-resources>
```

Cuối cùng là một trong nhiều ví dụ từ dự án Apache Torque. Đây là một nền tảng Java cung cấp một lớp ánh xạ đối tượng đến các quan hệ, cho phép tạo ra mã nguồn sử dụng các đối tượng Java để tạo ra các câu lệnh INSERT, UPDATE, DELETE, và SELECT tự động. Thông qua nền tảng này để kết nối với cơ sở dữ liệu, cần thực hiện một định nghĩa trong tệp torque.properties như dưới đây (ví dụ dưới đây là truy nhập vào CSDL MySQL):

```
torque.database=mysql
torque.database.url=jdbc:mysql:192.168.1.33/mysql
torque.database.driver=org.gjt.mm.mysql.Driver
torque.database.user=root
torque.database.password=rootpwd
torque.database.host=192.168.1.33
```

Điều đáng lo ngại là tất cả các ví dụ trên được lấy từ môi trường chính thống và thường là thiết lập mặc định của máy chủ và các mô hình ứng dụng. Bởi vì mục đích của các môi trường ứng dụng là giảm bớt các công việc cần thực hiện cho người phát triển ứng dụng. Do đó, những người phát triển thường không quan tâm đến cách thức thực hiện của môi trường mà chỉ tìm cách đạt được mục đích của họ càng nhanh càng tốt. Cho nên thường họ sẽ để mặc định các thiết lập này, dẫn đến các lỗ hổng là điều không thể tránh khỏi. Lưu ý rằng, mặc dù tất cả các ví dụ đã trình bày ở trên được lấy từ môi trường JAVA, tuy nhiên JAVA không phải là nguồn gốc của vấn đề. Ví dụ sau đây sẽ chỉ ra một nguồn dữ liệu sử dụng OLE DB trong môi trường kết nối Microsoft ADO tới SQL Server, chuỗi kết nối này là một điển hình và đôi khi được lưu trữ trong tệp rõ của máy chủ ứng dụng:

```
Provider=SQLOLEDB;  
Data Source=192.168.1.32;  
Initial Catalog=Northwind;  
User ID=sa;  
Password=sapwd;
```

Vì sự lười biếng của các nhà phát triển, họ thường có xu hướng lưu các đoạn mã ngắn ở đâu đó trong thư mục gốc của họ, và những đoạn mã này thường chứa những mật khẩu CSDL trong đó. Chẳng hạn có một môi trường SQL đã được quản lý để áp dụng chính sách mật khẩu mạnh, thay vì kết nối với CSDL sử dụng lệnh mysql -uroot -proot <dbname>, một nhà phát triển có thể dùng mysql -udev-pG7jds89krt <dbname>. Trong trường hợp này, nhiều nhà phát triển sẽ tạo ra một đoạn mã thực thi trong thư mục gốc của họ (hoặc đường dẫn tương ứng) dưới một cái tên ngắn:

```
mysql -udev -pG7jds89krt <dbname>
```

Do đó, tất cả những gì một kẻ tấn công cần làm là tấn công vào máy tính của nhà phát triển và tìm kiếm đoạn mã đó. Do các nhà phát triển thường chủ

quan nên kẻ tấn công có thể dễ dàng thực hiện tìm kiếm (ví dụ) “mysql -u” sử dụng lệnh tìm kiếm và lọc (find và grep nếu ở trong môi trường unix).

Sau khi chiếm quyền quản lý máy tính của nhà phát triển, kẻ tấn công có thể xem danh sách các tiến trình. Một số ứng dụng không làm ẩn đi các câu lệnh mà người dùng thực thi, ví dụ, nhà phát triển sử dụng lệnh tsql để kết nối tới Sysbase hoặc tới máy chủ CSDL từ một máy linux sử dụng lệnh :

```
Tsql -H falcon.guardium.com -p 1433 -U sa -P sapwd
```

Kẻ tấn công có thể dùng lệnh:

```
Ps auxwwg | grep tsql
```

Lệnh ps với tham số lọc “tsql” sẽ hiện ra tất cả các tiến trình có tên “tsql”, từ đó có thể xem toàn bộ câu lệnh được thực thi là gì, kết quả thu được:

```
ronb 16193 0.0 0.3 6616 2044 pts/5 T 11:05 0:00 lttsql -H  
falcon.guardium.com -p 1433 -U sa -P sapwd
```

Trong ví dụ này chỉ lấy ví dụ về việc kẻ tấn công khám phá được mật khẩu một cách đơn giản. Kẻ tấn công có thể cài đặt một đoạn mã thực thi để lặp lại quá trình tìm kiếm, cứ mỗi lần tìm kiếm tới dòng thông tin trên sẽ lưu vào một tập tin ẩn, như vậy kẻ tấn công hoàn toàn có thể khai thác tấn công này trong thực tế. Tuy nhiên, hiện nay các công cụ mới đã có những biện pháp để ẩn một số dòng lệnh khỏi lệnh ps. Ví dụ sau cho thấy kết quả truy vấn từ Oracle's plsql, Sybase's isql, và MySQL's mysql, trong tất cả các trường hợp, các mật khẩu không được hiển thị ngay cả khi các dòng lệnh được nạp vào chương trình như một tham số.

```
ronb 16249 0.6 0.7 7640 3608 pts/5 S 11:04 0:00
mysql -uroot -pxxxxxxxxxx DB
ronb 16253 0.1 0.9 12684 5060 pts/5 S 11:06 0:00
sqlplus
ronb 16256 0.0 0.2 2736 1424 pts/5 S 11:07 0:00
isql -Usa -S eagle
```

4.2.2. Kiểm soát việc truy nhập vào CSDL

Bước đầu tiên trong việc giải quyết các lỗ hổng bảo mật về mật khẩu CSDL là giám sát xem ai đang truy nhập vào dữ liệu. Bắt đầu bằng việc tạo báo cáo về các tài khoản nào đang hoạt động, tài khoản đó đang dùng địa chỉ IP nào kết nối tới CSDL, và ứng dụng nào được dùng để truy nhập vào CSDL, hình 4.2 là một ví dụ về một báo cáo (tên người dùng đã được làm mờ để tránh rò rỉ thông tin cho kẻ tấn công)

data access baseline			
DB User Name	Client IP	Source Program	Count of Client/Serv
wsoptser	192.168.67.72	C:\Program Files\Editorial_bgnd\cciprog\OraDriver.exe	1
wsoptser	198.115.79.198	C:\Program Files\Editorial_bgnd\cciprog\OraDriver.exe	1
wmsi_test	198.115.79.133	C:\WINNT\Microsoft.NET\Framework\v1.1.4322\aspnet_wp.exe	2
wms	192.168.66.27	C:\WINNT\Microsoft.NET\Framework\v1.1.4322\aspnet_wp.exe	1
wms	198.115.66.232	F:\PROJECTS\WMS\Utilities\bin\Debug\Utilities.exe	1
White	198.115.68.207		3
White	198.115.68.207	Microsoft? Access	2
wfdd	192.168.77.45	C:\Program Files\Editorial_bgnd\cciprog\OraDriver.exe	1
WULTERMAN	206.33.106.184		1
t_coakley	192.168.67.31	C:\Program Files\Editorial_bgnd\cciprog\OraDriver.exe	1
t_coakley	198.115.71.19	C:\Program Files\Editorial_bgnd\cciprog\OraDriver.exe	1
TU	198.115.68.221		6
TU	198.115.68.221	Microsoft? Access	3
TOOTSIAN	198.115.75.203		6
TOOTSIAN	198.115.75.203	Microsoft? Access	4
S_Mughan	192.168.68.19		2
s_dolaney	198.115.67.199	C:\Program Files\Editorial_bgnd\cciprog\OraDriver.exe	1
s_waterson	198.115.67.241	C:\Program Files\Editorial_bgnd\cciprog\OraDriver.exe	1
swat?	198.115.79.245	C:\Program Files\Editorial_bgnd\cciprog\OraDriver.exe	1
Subsonberlin	192.168.11.14	Internet Information Services	13
Subsonberlin	192.168.11.14	JavaService	2
Subsonberlin	192.168.11.14	Microsoft(R) Windows (R) 2000 Operating System	29
Subsonberlin	192.168.11.14	Service Runner	23
Subsonberlin	192.168.11.15	Internet Information Services	17
Subsonberlin	192.168.11.15	Microsoft(R) Windows (R) 2000 Operating System	16
Subsonberlin	192.168.11.15	Service Runner	21
Subsonberlin	192.168.66.154	Internet Information Server	1
Subsonberlin	192.168.66.154	JavaService	2
Subsonberlin	192.168.69.103	Microsoft (r) Windows Script Host	4
Subsonberlin	198.115.87.22		2
storefront	198.115.68.52	Microsoft (R) .NET Framework	9
stateapp	198.115.66.29	Internet Information Services	1
SSIV	192.168.68.64		1
SQAdmin	192.168.68.68		5
SQAdmin	192.168.68.68	Microsoft? Access	3
SoftwareShop	192.168.68.19	Layout Application	4
SoftwareShop	192.168.75.41	SCOR	1
SoftwareShop	198.115.69.124		4
SoftwareShop	198.115.69.22	SCOR	1
Skoletsky	198.115.75.32		4
Skoletsky	198.115.75.32	Microsoft? Access	1
Skoletsky	198.115.75.41		2
skierett2	198.115.79.161	C:\Program Files\Editorial_bgnd\cciprog\OraDriver.exe	1

Hình 4.2 Liệt kê danh sách các tài khoản đang dùng để truy nhập CSDL và chương trình được sử dụng

Báo cáo này có chỉ ra nhiều thông tin quan trọng:

1. Báo cáo này cho thấy ai đang truy nhập vào cơ sở dữ liệu, thông tin này được dùng để tìm ra người chủ của ứng dụng và xem xét mật khẩu người này được lưu trữ ở máy khách như thế nào, có an toàn không. Việc làm này có thể gặp khó khăn và mất nhiều thời gian vì đôi khi việc giám sát này phải làm việc với những người không nằm trong nhóm, trong cùng phòng ban.

2. Sau khi đã lập danh sách các điểm truy cập bình thường, ghi vào trong báo cáo có thể gọi là **mẫu cơ sở** (mẫu các hoạt động thông thường) so sánh các báo cáo mới với mẫu bình thường để tìm và phát hiện ra các điểm truy

nhập mới, hoặc tạo ra một cảnh báo tức thời để thông báo cho người quản trị biết một điểm truy nhập mới vừa xuất hiện. Một điểm truy nhập mới được phát hiện có thể là:

- *Trường hợp thứ nhất*: đây có thể là một ứng dụng mới, máy tính mới hợp pháp, được phép truy nhập vào CSDL. Ví dụ như khi nâng cấp các trình điều khiển CSDL, máy chủ ứng dụng, các công cụ, module hoặc phần mềm mới được cài đặt. Trong các trường hợp, nên xem xét lại các điểm truy nhập để đảm bảo không mắc phải các vấn đề bảo mật như mật khẩu truyền ở dạng rõ. Ngoài ra khi phát hiện một tài khoản người dùng đang được dùng tại nhiều máy có địa chỉ IP khác nhau, cần thực hiện giới hạn cho chỉ một địa chỉ IP được sử dụng tại một thời điểm.
- *Trường hợp thứ hai*: có thể là tấn công thật, được thực hiện từ các kẻ tấn công. Khi họ có được tên người dùng và mật khẩu từ các ứng dụng, họ thường kết nối tới CSDL từ các máy tính khác nhau, sử dụng các chương trình khác nhau. Ví dụ, kẻ tấn công có thể chạy một đoạn mã (script) từ máy laptop của anh ta; điều này dễ dàng hơn nhiều so với việc xâm nhập vào một máy chủ ứng dụng để chạy các câu lệnh. Hơn nữa kẻ tấn công có nguy cơ bị phát hiện nếu anh ta đăng nhập vào máy chủ dịch vụ trong một thời gian dài. Và việc đơn giản và an toàn hơn hết là lấy được tên và mật khẩu người dùng để tiếp tục tấn công từ máy tính của kẻ tấn công. Điều này nghĩa là, bằng cách giám sát truy nhập dữ liệu, chúng ta có thể xác định được cuộc tấn công. Do đó, trong một môi trường ổn định và có ít hoặc thường không có thay đổi gì so với **mẫu cơ sở** thì một cảnh báo tức thời là rất cần thiết khi có sự kiện bất thường xảy ra.

Cách đơn giản nhất để tạo ra các báo cáo là sử dụng công cụ bảo mật CSDL của bên thứ ba, có thể tìm kiếm với các cụm từ “who-what-when-where” và các từ liên quan tới truy nhập CSDL hoặc kiểm toán CSDL. Những

sản phẩm này thường sẽ có tính năng báo cáo cho phép chúng ta dễ dàng tạo, sửa, xóa báo cáo.

Một cách khác không cần sử dụng công cụ từ bên thứ ba là tự tạo ra báo cáo. Tuy nhiên việc này sẽ khó khăn hơn rất nhiều vì báo cáo tự tạo chỉ có thể thực hiện được tại thời điểm kiểm tra hệ thống, không hỗ trợ tức thời (thời gian thực) khi có sự kiện bất thường xảy ra, không hỗ trợ tạo ra **mẫu cơ sở**, và không thể triển khai trên hệ thống lớn.

Ví dụ, để có được thông tin truy nhập vào Oracle, có thể thực hiện truy vấn v\$session như sau:

select machine, terminal, program from v\$session;		
Trả về kết quả		
USERNAME	MACHINE	PROGRAM
-----	-----	-----
SYSTEM	WORKGROUP\RON-NY	sqlplusw.exe

Máy khách tên “RON-NY” truy nhập vào CSDL sử dụng lệnh sqlplusw.exe truy nhập vào CSDL. Có thể dùng lệnh sau để tìm các thông tin liên quan:

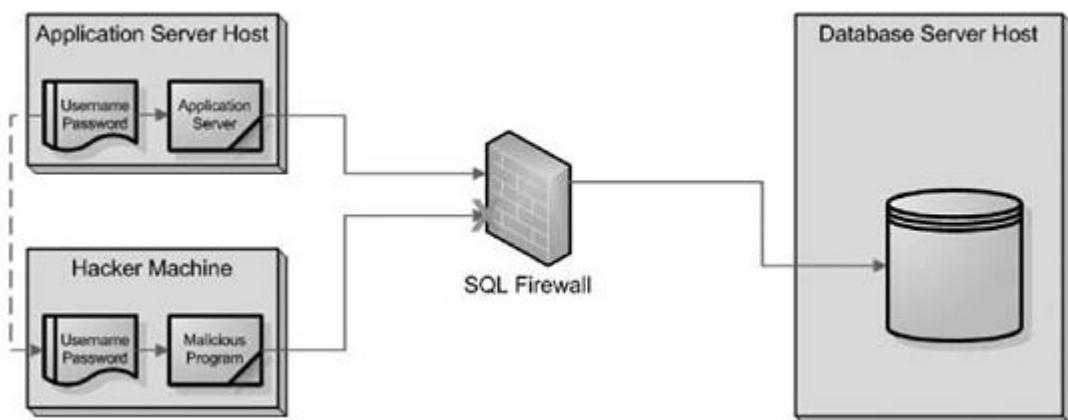
```
select loginame, hostname, program_name from sysprocesses
```

Với cả hai trường hợp trên việc thực hiện giám sát sẽ phải liên tục ghi ra những thông tin thu thập được, rồi từ đó tạo ra một **mẫu cơ sở**. Hoặc có thể sử dụng chức năng kiểm toán của CSDL để tạo ra mẫu sự kiện thông thường này.

Khi đã có một **mẫu cơ sở**, có thể thực hiện ngăn chặn các truy nhập vào CSDL không có trong mẫu. Quay lại với trường hợp khi kẻ tấn công lấy cắp tài khoản và mật khẩu người dùng từ tệp cấu hình trên các máy chủ CSDL và kết nối tới CSDL từ chính máy của mình. Trường hợp này, ta có thể dễ dàng phát hiện ra có một địa chỉ IP mới không có trong **mẫu cơ sở**, ta có thể chặn

tấn công này bằng cách chặn địa chỉ IP “lạ” đó tới CSDL, việc này có thể được thực hiện bởi tường lửa.

Một tùy chọn thêm để tăng cường an toàn là sử dụng tường lửa, như trong hình 4.3, tại đây có hai lựa chọn: (1) sử dụng tường lửa tiêu chuẩn, cho phép thực hiện kiểm soát truy nhập dựa trên IP và Port; (2) sử dụng tường lửa SQL, cho phép việc xây dựng các tập luật không chỉ dựa trên IP mà còn dựa trên tên truy nhập, ứng dụng truy nhập, đối tượng CSDL được truy nhập. Nó cũng cho phép xác định chính xác tài khoản nào hay ứng dụng nào, máy tính nào được phép truy nhập vào CSDL.



Hình 4.3 Sử dụng một tường lửa SQL giữa ứng dụng và cơ sở dữ liệu

Ngoài ra chúng ta có thể kết hợp sử dụng thêm một phần mềm cảnh báo thời gian thực khi có bất kỳ vi phạm an ninh nào xảy ra. Kẻ tấn công có thể cố gắng kết nối tới CSDL từ máy tính của anh ta, và khi không thể kết nối được do bị tường lửa SQL chặn thì anh ta có thể đoán rằng địa chỉ IP này đang bị chặn (đang có một thiết bị giám sát an ninh tầng 3). Kẻ tấn công có thể quay ra thực hiện tấn công các máy chủ ứng dụng, hoặc có thể giả mạo địa chỉ IP của máy chủ ứng dụng thành địa chỉ IP của máy mình và tiếp tục tấn công. Quá trình lọc có thể mất thêm thời gian, tuy nhiên nếu thực hiện được việc lọc bằng cách sử dụng phần mềm cảnh báo thời gian thực thì có thể chặn cuộc tấn công từ sớm, trước khi kẻ tấn công tìm mọi cách để vượt qua các hàng rào bảo vệ.

Một tường lửa SQL là cách duy nhất để thực thi cơ chế kiểm soát truy nhập, đặc biệt là đối với cơ sở dữ liệu phức tạp, khó quản trị. Sử dụng tường lửa SQL, cho phép định nghĩa cẩn thận các luật, xác định những gì được phép và không được phép truy nhập, các mức độ truy nhập, ứng dụng truy nhập, xác định IP, câu lệnh được thực thi, vv... CSDL thông thường không thể cung cấp mức độ kiểm soát truy nhập như vậy.

Một chức năng được biết đến đó là SQL*Plus thực hiện chức năng kiểm soát truy nhập trong CSDL, nó cho phép giới hạn truy nhập dựa trên tài khoản đăng nhập và quyền của tài khoản đó đối với CSDL. Nó chỉ ra các câu lệnh được phép hoặc không được phép thực hiện. Chức năng này được hỗ trợ thông qua việc sử dụng các bảng PRODUCT_PROFILE (và thông qua việc xem PRODUCT_PRIVS được sử dụng bởi những người dùng khác trên hệ thống) :

Name	Null?	Type
PRODUCT	NOT NULL	VARCHAR2(30)
USERID		VARCHAR2(30)
ATTRIBUTE		VARCHAR2(240)
SCOPE		VARCHAR2(240)
NUMERIC_VALUE		NUMBER(15,2)
CHAR_VALUE		VARCHAR2(240)
DATE_VALUE		DATE
LONG_VALUE		LONG

Khi thực hiện đăng nhập vào CSDL sử dụng SQL*Plus, công cụ này sẽ đưa ra các truy vấn sau tới CSDL:

```

SELECT
ATTRIBUTE,SCOPE,NUMERIC_VALUE,CHAR_VALUE,DATE_VA
LUE FROM

SYSTEM.PRODUCT_PRIVS

WHERE
(UPPER('SQL*Plus')      LIKE      UPPER(PRODUCT))    AND
(UPPER(USER) LIKE USERID)

```

Nếu sử dụng câu lệnh sau:

```
insert into  
product_profile(product, userid, attribute, char_value)  
values('SQL*Plus', 'SCOTT', 'UPDATE', 'DISABLED');
```

Và tiếp theo đăng nhập vào dưới tài khoản scott, thì sau đó khi cô gắng thực hiện cập nhật (dùng lệnh update) thông qua SQL*Plus sẽ cho thông báo lỗi sau:

SP2-0544: invalid command: update

Đây là một chức năng bảo mật rất hữu ích nhưng thật không may PRODUCT_PROFILE chỉ hoạt động với SQL*Plus. Nếu chúng ta cần chức năng này, thì phải dùng tường lửa SQL.

Kỹ thuật cuối cùng có thể giúp giải quyết vấn đề mật khẩu lưu dưới dạng rõ, dễ bị lộ khi truyền đi trong quá trình xác thực là sử dụng máy chủ LDAP để lưu trữ tất cả các tên người dùng và mật khẩu và LDAP dùng để xác thực tài khoản cho máy chủ ứng dụng và máy chủ CSDL. Nhiều khả năng mật khẩu sẽ không được lưu tại các tệp cấu hình trong CSDL nữa, bởi đa phần các CSDL hiện nay đều hỗ trợ cơ chế xác thực này (và khuyến nghị việc nên sử dụng hình thức xác thực này), chúng ta nên cân nhắc để sử dụng sao cho phù hợp với nhu cầu của mình. Chú ý rằng khi triển khai, nó tạo ra một môi trường bảo mật hơn, nhưng đôi khi nó chỉ làm thay đổi các điểm dễ bị tấn công (ví dụ kẻ tấn công có thể tấn công vào máy chủ LDAP để khai thác tài khoản mật khẩu truy nhập vào chính máy chủ LDAP, nên xem xét cấu hình LDAP xem tài khoản và mật khẩu có dễ bị lộ không).Thêm nữa là có thể sử dụng kỹ thuật này song song với các cơ chế kiểm soát truy nhập, và giám sát truy nhập sử dụng **mẫu cơ sở** hay triển khai các chính sách sử dụng tường lửa SQL để tăng cường bảo mật.

4.3 XÁO TRỘN MÃ ỦNG DỤNG

Một loại lỗ hổng bảo mật trong ứng dụng phổ biến ngày nay là mã nguồn ứng dụng thường quá lộ liễu, dễ tiếp cận. Tùy thuộc vào ngôn ngữ lập trình

được sử dụng để viết nên ứng dụng, một kẻ tấn công có thể trích xuất mã nguồn để phát hiện xem làm thế nào để ứng dụng đó có thể truy nhập vào cơ sở dữ liệu. Điều này giúp ích rất nhiều cho việc tấn công lên CSDL, bằng cách trực tiếp hay thông qua ứng dụng đó, xem thêm hình 4.3.

4.3.1 Phân tích điểm yếu mã nguồn và giả mã

Trong nhiều môi trường ứng dụng hiện đại ngày nay, một số mã được đặt trong định dạng mã nguồn hoặc trong định dạng mà có thể dễ dàng được sử dụng cũng như dễ dàng lấy được. Và từ đó kẻ tấn công có thể có được rất nhiều thông tin về các hoạt động bên trong của ứng dụng (đây cũng là một vi phạm về quyền sở hữu trí tuệ, nhưng không được đề cập đến trong chương này). Một kẻ tấn công có thể tìm hiểu các cách thức kết nối trong ứng dụng với CSDL, hay làm thế nào tên tài khoản được mã hóa, cũng như các câu truy vấn tới CSDL mà ứng dụng sử dụng là gì. Mã nguồn cũng có thể bị kẻ tấn công phân tích để tìm ra những lỗ hổng trong ứng dụng.

Mã ứng dụng được viết dưới định dạng mã nguồn phổ biến ở cả Java và Microsoft. Trong môi trường Java một số mã được lưu trữ như Java Server Page (JSP) và trong môi trường Microsoft như Active Server Page (ASP), cả hai định dạng này chủ yếu được sử dụng để xử lý lớp “trình diễn” (presentation layer). Những tập tin này thường trông như các file HTML với các mã nhúng hoặc các đoạn “codebehind”, Ví dụ ba danh sách sau đây cho thấy một đoạn JSP bao gồm mã nhúng, một đoạn aspx được sử dụng để tạo ra các trang web, và thành phần liên quan, aspx_cs mã C# codebehind. Lưu ý rằng tất cả các thành phần này thường được viết như mã nguồn trên các máy chủ dịch vụ:

JSP Fragment:

```
<table border="0" align="left">
<tr>
<td class="SubHeading" colspan="9">
<span class="MultiLingual" id="170">
Report Additional Part Usage
</span>
</td>
</tr>
<% List actualParts = bean.getActualParts();
int size1 = 0;
if (actualParts != null)
size1 = actualParts.size();
for (int j=0; j<size1; j++){
Map tmpPartsMap = (HashMap)actualParts.get(j);
String qtyAvailable =
(String)tmpPartsMap.get("qtyAvailable");
String qtyUsed = (String)tmpPartsMap.get("qtyUsed");
%>
<tr>
<td>
<input type="checkbox" name="stamBill<%=j%>" disabled
<% if (tmpPartsMap.get("billable").equals("true")){ %>
checked
<% }%>
</td>
<% int numberOfRows = bean.getNumberOfEmptyRows();
List savedAddParts = bean.getSavedAddParts();
```

```

if (savedAddParts == null)
    numberOfRows = 0;
for (int k=0; k<numberOfRows; k++){
    Map savedValues = (HashMap)savedAddParts.get(k);
    %>
    <tr valign="top">
        <TD>
            <SELECT class="FreeText" NAME="stockTrans<%=k%>" 
                onChange="fillBillable(<%=k%>)" SIZE=1>
            <% List stockTrans = bean.getStockTransactions();
            int size2 = 0;
            if (stockTrans != null)
                size2 = stockTrans.size();
            for (int i=0;i < size2; i++) {
                Map map = (HashMap)stockTrans.get(i);
            %>

```

aspx Fragment:

```

<asp:TextBox MaxLength="50" id="FirstName" runat="server"
/>
<asp:RequiredFieldValidator ControlToValidate="FirstName"
Display="dynamic" Font-Name="verdana" Font-Size="9pt"
ErrorMessage="First Name' must not be left blank."
runat="server" id="RequiredFieldValidator1"></
asp:RequiredFieldValidator>
<asp:TextBox MaxLength="50" id="LastName" runat="server"
/>
<asp:RequiredFieldValidator ControlToValidate="LastName"

```

```

Display="dynamic" Font-Name="verdana" Font-Size="9pt"
ErrorMessage="Last Name' must not be left blank."
runat="server" id="RequiredFieldValidator5">></
asp:RequiredFieldValidator>
<asp:TextBox MaxLength="50" id="Email" runat="server" />
<asp:RegularExpressionValidator ControlToValidate="Email"
ValidationExpression="[\w\.-]+(\+[\w-]*\)?@([\w-]+\.)+[\w-]+"
Display="Dynamic" Font-Name="verdana" Font-Size="9pt"
ErrorMessage="Must use a valid email address." runat="server"
id="RegularExpressionValidator1">></
asp:RegularExpressionValidator>
<asp:RequiredFieldValidator ControlToValidate="Email"
Display="dynamic" Font-Name="verdana" Font-Size="9pt"
ErrorMessage="Email' must not be left blank." runat="server"
id="RequiredFieldValidator2">></asp:RequiredFieldValidator>
<asp:TextBox      MaxLength="25"      id="Password"
TextMode="Password"
runat="server" />
<asp:RequiredFieldValidator ControlToValidate="Password"
Display="dynamic" Font-Name="verdana" Font-Size="9pt"
ErrorMessage="Password' must not be left blank."
runat="server" id="RequiredFieldValidator3">></
asp:RequiredFieldValidator>
<asp:TextBox MaxLength="25" id="ConfirmPassword"
TextMode="Password" runat="server" />
<asp:RequiredFieldValidator
ControlToValidate="ConfirmPassword"      Display="dynamic"
FontName="verdana"      Font-Size="9pt"      ErrorMessage="Confirm'
must not
be left blank." runat="server" id="RequiredFieldValidator4">></

```

```
asp:RequiredFieldValidator>
<asp:CompareValidator ControlToValidate="ConfirmPassword"
ControlToCompare="Password" Display="Dynamic"
FontName="verdana" Font-Size="9pt" ErrorMessage="Password
fields do
not match." runat="server" id="CompareValidator1"></
asp:CompareValidator>
<asp:ImageButton id="RegisterButton" ImageUrl="images/
submit.gif" runat="server" />
```

aspx_cs fragment:

```
public class Register : System.Web.UI.Page {
protected System.Web.UI.WebControls.TextBox FirstName;
protected System.Web.UI.WebControls.TextBox LastName;
protected System.Web.UI.WebControls.TextBox Password;
protected System.Web.UI.WebControls.TextBox
ConfirmPassword;
protected System.Web.UI.WebControls.CompareValidator
CompareValidator1;
protected System.Web.UI.WebControls.Label MyError;
protected System.Web.UI.WebControls.ImageButton
RegisterButton;
private void RegisterButton_Click(
object
sender, System.Web.UI.ImageClickEventArgs e) {
```

```

if (Page.IsValid == true) {
    ShoppingCartDB shoppingCart = new
    ShoppingCartDB();
    String tempCartId =
        shoppingCart.GetShoppingCartId();
    CustomersDB accountSystem = new CustomersDB();
    try {
        String customerId =
            accountSystem.AddCustomer(FirstName.Text,
        LastName.Text,
        Email.Text, Password.Text);
        FormsAuthentication.SetAuthCookie(customerId, false);
        shoppingCart.MigrateCart(tempCartId, customerId);
        Response.Cookies["AdventureWorks_FullName"].Value
        = Server.HtmlEncode(FirstName.Text + " " +
        LastName.Text);
        Response.Redirect("ShoppingCart.aspx");
    } catch (UserAlreadyExistsException) {
        MyError.Text = "Registration failed:  That email
        address is already registered.<br><img align=left height=1
        width=92 src=images/1x1.gif>";
    }
}
}
}

```

JSPs và ASPs được triển khai rất nhiều cho việc phát triển ứng dụng. Điều này đúng ngay cả trên hệ thống sản xuất và các hệ thống công cộng trên internet. Mặc dù các máy chủ ứng dụng sẽ lưu trữ những tập tin trong thư mục mà người dùng không thể truy nhập vào, nhưng đã có rất nhiều trường hợp phát hiện lỗ hổng bảo mật trên Web và máy chủ ứng dụng cho phép truy

nhập được vào những tập tin này. Ví dụ, Sun alert ID 55221 (6/2003) cảnh báo một lỗi của JSPs được sử dụng trong máy chủ ứng dụng Sun ONE cho phép xem được mã nguồn, và Oracle security alert #47 (12/2002) báo cáo rằng máy chủ ứng dụng Oracle9i phiên bản 9.0.2.0.0 cho phép kẻ tấn công từ xa có thể lấy được mã nguồn từ các tập tin JSP bằng cách gửi một yêu cầu URL đặc biệt chỉ ra một JSP cụ thể, kết quả là truy vấn trả về mã nguồn của tập tin thay vì truy vấn được xử lý.

Một vấn đề khác về mã bảo vệ liên quan đến mã giả hoặc định dạng trung gian. Cả hai môi trường Java và Microsoft .Net đều dựa trên một máy ảo (VM) mẫu, trong đó mã nguồn được biên dịch thành một định dạng trung gian (hay còn gọi là mã giả) mà sau đó được sử dụng bởi các máy ảo để chạy ứng dụng. Định dạng trung gian này bao gồm các hướng dẫn, chỉ thị cụ thể đối với máy ảo, máy ảo thực hiện những chỉ thị này và đổi khi biên dịch chúng thành mã máy (code on-the-fly / just in time compile). Lợi thế của kiến trúc này là tính di động và khả năng tương tác tốt. Ví dụ trong môi trường Java cho phép một chương trình chạy trên một hệ điều hành có cài máy ảo Java (JVM - java virtual machine) mà không cần phải sửa đổi hay biên dịch lại code. Các lớp (class) trong Java được biên dịch thành các file .class chứa các bytecode, chỉ thị, điều khiển máy ảo java (JVM) . Nền tảng .Net thực hiện các tương tác với Microsoft, trong đó nhiều ngôn ngữ lập trình có thể tương tác tốt và chia sẻ các tập lệnh dễ dàng bởi tất cả đều được biên dịch theo một định dạng chung trên một cơ sở chung. Cơ sở chung này được gọi là Common Library Runtime(CLR), và các định dạng trung gian trong đó tất cả các chương trình được biên dịch sẽ được gọi là Common Intermediate Language(CIL). Định dạng CIL bao gồm siêu dữ liệu (metadata) có thể đọc hiểu code và cung cấp nhiều thông tin có giá trị cho hacker.

Trong khi máy ảo và mã giả rất thuận tiện cho việc phát triển và đơn giản hóa việc triển khai phần mềm, thì chính nó cũng là phần dễ bị tổn thương. Mã giả gồm những tập lệnh chỉ thị ở mức máy ảo, và máy ảo chạy như một phần mềm thông dịch, nó chuyển đổi mã nguồn của một ngôn ngữ lập trình thành các lệnh cho máy tính thực hiện tính toán. Do những tập lệnh này rất thông

dụng nên kẻ tấn công đã xây dựng các chương trình gọi là phần mềm lây cắp mã giả. Các chương trình này đọc biên soạn mã giả và tạo ra mã nguồn. Mã nguồn này gần chính xác như mã gốc. Trong thực tế thì mã nguồn này chỉ hơi khác một chút là không có chú thích, giải thích các lệnh. Ví dụ, danh sách sau được tạo ra từ một lớp tập tin Java (một số method đã được rút gọn):

```
// Decompiled by Jad v1.4.8f. Copyright 2001 Pavel Kouznetsov.
// Jad home page: http://www.kpdus.com/jad.html
// Decompiler options: packimports(3)
// Source File Name: Errors.java
package sqlj.runtime.error;
import java.sql.SQLException;
import java.text.MessageFormat;
import java.util.MissingResourceException;
import java.util.ResourceBundle;
public class Errors
{
    public Errors()
    {
    }
    public static void raiseError(String s, ResourceBundle resourcebundle, String s1)
        throws SQLException
    {
        raiseError(s, resourcebundle, s1, new Object[0]);
    }
    ...
}
```

```
String s1,
Object aobj[])
throws SQLException
{
throw new SQLException(getText(resourcebundle, s1, aobj), s);
}
public static String getText(ResourceBundle resourcebundle, String
s)
{
return getText(resourcebundle, s, new Object[0]);
}
public static String getText(ResourceBundle resourcebundle, String
s, Object obj)
{
return getText(resourcebundle, s, new Object[] {
obj
});
}
...
public static String getText(ResourceBundle resourcebundle, String
s, Object
aobj[])
{
if(resourcebundle == null)
return "unable to load resource bundle for message key " + s;
try
{
return MessageFormat.format(resourcebundle.getString(s), aobj);
}
}
```

```

        catch(MissingResourceException missingresourceexception)
    {
        return "unable to find error message for key " + s;
    }
}

public static final String DEFAULT_SQLSTATE = "46000";
public static final String
UNSUPPORTED_FEATURE_SQLSTATE = "46110";
public static final String
INVALID_CLASS_DECLARATION_SQLSTATE = "46120";
public static final String
INVALID_COLUMN_NAME_SQLSTATE = "46121";
public static final String
INVALID_PROFILE_STATE_SQLSTATE = "46130";
}

```

Chú ý rằng tất cả các giá trị và các method có thể nhìn thấy được và nhược điểm duy nhất là các biến được đặt tên s, s1, obj. Tiện ích Disassembly có thể sử dụng lệnh javap trong JDK, nhưng nó chỉ cung cấp với disassembly ở mức tập lệnh cho máy ảo. Tiện ích này có thể được sử dụng để dịch ngược mã. Nhiều bản quyền chương trình không cho phép dịch ngược. Nếu chúng ta không phải là chủ sở hữu bản quyền của mã mà muốn dịch ngược, hãy kiểm tra các bản quyền xem có được phép dịch ngược mã hay không.

```

Compiled from Errors.java
public class sqlj.runtime.error.Errors extends java.lang.Object {
    public static final java.lang.String DEFAULT_SQLSTATE;
    public static final java.lang.String
UNSUPPORTED_FEATURE_SQLSTATE;
    public static final java.lang.String

```

```
INVALID_CLASS_DECLARATION_SQLSTATE;
public static final java.lang.String
INVALID_COLUMN_NAME_SQLSTATE;
public static final java.lang.String
INVALID_PROFILE_STATE_SQLSTATE;
public sqlj.runtime.error.Errors();
public static void raiseError(java.lang.String,
java.util.ResourceBundle,
java.lang.String) throws java.sql.SQLException;
public static void raiseError(java.lang.String,
java.util.ResourceBundle,
java.lang.String, java.lang.Object) throws java.sql.SQLException;
public static void raiseError(java.lang.String,
java.util.ResourceBundle,
java.lang.String, java.lang.Object, java.lang.Object) throws
java.sql.SQLException;
public static void raiseError(java.lang.String,
java.util.ResourceBundle,
java.lang.String, java.lang.Object, java.lang.Object,
java.lang.Object) throws
java.sql.SQLException;
public static void raiseError(java.lang.String,
java.util.ResourceBundle,
java.lang.String, java.lang.Object[]) throws
java.sql.SQLException;
public static java.lang.String getText(java.util.ResourceBundle,
java.lang.String);
```

```
public static java.lang.String getText(java.util.ResourceBundle,  
java.lang.String, java.lang.Object);  
public static java.lang.String getText(java.util.ResourceBundle,  
java.lang.String, java.lang.Object, java.lang.Object);  
public static java.lang.String getText(java.util.ResourceBundle,  
java.lang.String, java.lang.Object, java.lang.Object,  
java.lang.Object);  
public static java.lang.String getText(java.util.ResourceBundle,  
java.lang.String, java.lang.Object[]);  
}
```

4.3.2 Tiền biên dịch và mã hóa mã ứng dụng

Một phương pháp để hạn chế các tấn công khai thác lỗ hổng dựa trên phân tích mã nguồn là sử dụng các kỹ thuật mã hóa ứng dụng, phần này sẽ trình bày một số kỹ thuật cơ bản.

Do JSP kết hợp cả mã của HTML và Java, nên chúng không chạy máy chủ ứng dụng Java. Khi JSP chạy thì trước tiên, máy chủ ứng dụng Java phải qua một quá trình biên dịch. Máy chủ đọc mã nguồn JSP và chuyển nó thành một class Java được gọi là một servlet. Quá trình này tạo ra một file .java, được lưu trữ trên hệ thống. Sau đó, máy chủ biên dịch những lớp Java vừa được tạo này để tạo file .class, mà sau đó được dùng để xử lý các tiến trình. Quá trình dịch và biên dịch thuận lợi cho các nhà phát triển. Lưu ý rằng quá trình này để lộ ra mã nguồn hai lần; một là JSP và một là mã nguồn Java.

Để giải quyết vấn đề tấn công phân tích mã nguồn JSP, tại thời điểm triển khai cần thực hiện chuẩn bị tất cả các tệp class từ JSPs. Nếu máy chủ đã có các tệp class mới nhất, nó sẽ bỏ qua các bản dịch và giai đoạn biên dịch. Ví dụ, chúng ta có thể tải về một tác vụ Ant gọi là jspc để thực hiện tiến trình này từ <http://ant.apache.org/manual/OptionalTasks/jspc.html>. Một số máy chủ cũng cung cấp các tiện ích sẵn có và các hướng dẫn cho quá trình này, ví dụ, Oracle

9iAS có một tiện ích ojspc và Web Logic có lớp Java tích hợp sẵn gọi là weblogic.jspc

Bây giờ chúng ta bàn đến mã hóa ứng dụng, mã hóa ứng dụng là một kỹ thuật được sử dụng trong cả Java và .Net, nhằm chuyển các mã nguồn thành một dạng mà rất khó để có thể dịch ngược lại được (làm thay đổi mã nguồn trong khi giữ nguyên chức năng, để che giấu cấu trúc, mục đích, hay thuật toán).

Mã hóa ứng dụng bao gồm nhiều bước vì tất cả các thành phần của ứng dụng đều phải được mã hóa. Các loại mã hóa ứng dụng:

Mã hóa giao diện: bao gồm làm xáo trộn các nhận dạng, loại bỏ các chú thích câu lệnh, và đảo vị trí các method.

Mã hóa dữ liệu: ảnh hưởng tới cấu trúc dữ liệu và mã hóa dữ liệu. Ví dụ: một mảng hai chiều có thể được chuyển thành mảng một chiều và trải rộng.

```
int i=1;  
while (i < 100) {  
.. arr[i] ..  
i++  
}
```

Đoạn mã trên có thể được chuyển về dạng như sau:

```
int i = 7;  
..  
int j = i + 4;  
while (j < 8003) {  
.. arr[(j-3)/(i+1)] ..;  
j += (i+1);  
}
```

các
câu lệnh không liên quan.

- Thêm đoạn mã giúp chống lại việc biên dịch ngược. Ví dụ, trong nhiều năm, ứng dụng phổ biến nhất dùng cho việc dịch ngược mã Java là phần mềm freeware tên là Mocha. Một trình mã hóa giả được tạo ra gọi là HoseMocha đã gắn thêm các lệnh giả chống lại việc dịch ngược, mà không làm ảnh hưởng đến việc thực thi chương trình.

Dưới đây là một ví dụ đơn giản, hãng Sun Microsystems mã hóa thư viện core Java với giải pháp phòng ngừa DashO. Nếu thực hiện dịch ngược hoặc chạy javap on các lớp com.sun.security.x509.X509Key, sẽ nhận được lỗi sau:

Decompile:
Couldn't fully decompile method buildX509Key
Couldn't resolve all exception handlers in method buildX509Key

Javap:
Error: Binary file 'X509Key' contains
sun.security.x509.X509Key

Điều quan trọng là cần nhớ rằng các tiện ích mã hóa ứng dụng được phát triển như một freeware (phần mềm miễn phí), shareware (phần mềm chia sẻ), hay commercial product (sản phẩm thương mại). Những người quan tâm và chú trọng tới việc loại bỏ các lỗ hổng ứng dụng thì trong quá trình phát triển ứng dụng họ nên sử dụng những tiện ích này.

4.4 BẢO VỆ CƠ SỞ DỮ LIỆU KHỎI CÁC TẤN CÔNG SQL INJECTION

SQL injection là một kỹ thuật cho phép những kẻ tấn công lợi dụng lỗ hổng của việc kiểm tra dữ liệu đầu vào trong các ứng dụng web và các thông báo lỗi của hệ quản trị cơ sở dữ liệu trả về để inject (tiêm vào) và thi hành các câu lệnh SQL bất hợp pháp, SQL injection có thể cho phép những kẻ tấn công thực hiện các thao tác: delete, insert, update,... trên cơ sở dữ liệu của ứng dụng, thậm chí là máy chủ mà ứng dụng đó đang chạy, lỗi này thường xảy ra trên các ứng dụng web có dữ liệu được quản lý bằng các hệ quản trị cơ sở dữ liệu như: SQL Server, MySQL, Oracle, DB2, Sybase....

Tóm lại, SQL injection là một kỹ thuật được sử dụng dựa trên các lỗ và lỗ hổng bảo mật trong các lớp ứng dụng để thực hiện một cuộc tấn công vào cơ sở dữ liệu hoặc dữ liệu.

4.4.1 Hiểu biết về SQL injection

Hầu hết các ví dụ ở mục này sử dụng cú pháp của SQL Server. Tất cả các cơ sở dữ liệu có thể là mục tiêu của SQL injection. Trong thực tế, không phải tất cả các lỗ hổng đều nằm trong cơ sở dữ liệu, có thể lỗ hổng nằm trong lớp ứng dụng bên ngoài của cơ sở dữ liệu nhưng thời điểm đó các ứng dụng có kết nối tới cơ sở dữ liệu, khi đó cơ sở dữ liệu có thể là mục tiêu của các cuộc tấn công.

Bắt đầu với một ví dụ điển hình về xác thực ứng dụng. Giả sử ứng dụng có một form đăng nhập trong trình duyệt Web như trong hình 4.4:

The image shows a screenshot of a login form. On the left, there is a vertical bar with the text "Figure 5.4" and "Login form." above an arrow pointing right. The main area contains two input fields: "User ID:" and "Password:", both with empty text boxes. Below these fields is a link labeled "> ID & Password Help". At the bottom of the form are three buttons: "Log On", "Learn More", and "Enroll".

Hình 4.4 Form đăng nhập

Ứng dụng sau khi nhận được tên truy nhập và mật khẩu cần xác thực người dùng bằng cách kiểm tra sự tồn tại của người sử dụng trong bảng

USER và PASSWORD xem có khớp với dữ liệu trong cột PWD của bảng đó không.

Giả sử ứng dụng không xác thực hai trường user và password và các câu lệnh SQL được tạo ra bằng cách nối chuỗi, trong thực tế đoạn mã này là một ví dụ:

```
sqlString = "select USERID from USER where USERID  
= `" &  
    userId & "` and PWD = `" & pwd & "`"  
result = GetQueryResult(sqlString)  
If (result = "") Then  
    userHasBeenAuthenticated = False  
Else  
    userHasBeenAuthenticated = True  
End If
```

Trong đoạn mã này dữ liệu trong trường đầu vào là userID và pwd. Ứng dụng tạo ra các câu lệnh SQL bằng cách sử dụng câu lệnh SELECT.

Chúng ta thử xem những gì sẽ xảy ra nếu thực hiện gõ tên truy nhập và password như sau:

```
User ID: ` OR ``=`  
Password: ` OR ``=`
```

Trong trường hợp này sqlString được sử dụng tạo ra kết quả như sau:

```
select USERID from USER where USERID = `` OR ```` and PWD = ``  
OR ````
```

Phần này xem xét một số phương pháp khác được sử dụng trong cuộc tấn công SQL injection. Những phương pháp này đều dựa trên việc sử dụng nối chuỗi và không xác thực dữ liệu người dùng.

Như đã đề cập ở trên SQL injection không cụ thể cho bất kỳ loại cơ sở dữ liệu nào. Tuy nhiên, một số phương pháp tấn công cụ thể sử dụng hàm cơ sở dữ liệu cụ thể, thường dựa trên sự khác nhau giữa các loại cơ sở dữ liệu.

Ví dụ: Việc sử dụng /* trong SQL Server và */ trong MySQL.

Trong ví dụ về xác thực người dùng đã trình bày, với các ứng dụng xác thực mật khẩu có chiều dài hơn 12 ký tự sẽ làm cho injection ở trên thất bại. Trong trường hợp này quá trình tấn công có thể đặt ID và password người dùng như sau:

```
User ID: ` OR ``=`` --  
Password: abc
```

Vì bất kỳ lệnh nào sau dấu -- đều bị bỏ qua, việc chèn câu lệnh sẽ được thực hiện vào vị trí mật khẩu. Trong ví dụ sau, việc tấn công không phải để đăng nhập tìm kiếm thông tin mà sẽ thực hiện hành động phá hoại:

```
User ID: ` ; DROP TABLE USER ;--  
Password: abc  
or  
User ID: ` ; DELETE FROM USER WHERE ``=``  
Password: ` OR ``=``
```

Nó sẽ chuyển sang hai câu lệnh SQL Server sau:

```
select USERID from USER where USERID = ``;  
DROP TABLE USER ;--`  
and PWD = `` OR ``=``  
select USERID from USER where USERID = ``;  
DELETE FROM USER  
WHERE ``=`` and PWD = `` OR ``=``.
```

T
vào v
phụ thuộc
ng bởi các
ứng dụng. Một cách để tránh được tấn công loại này là không cấp quyền cho

việc sử dụng các lệnh Delete hoặc Drop. Tuy nhiên, trong một số tình huống dẫn đến một số bất tiện trong các ứng dụng.

Một trong những kỹ thuật sẽ được trình bày trong phần sau để thực hiện tấn công SQL injection là tìm kiếm các mẫu đôi khi được gọi là các dấu hiệu. Việc tìm kiếm các mẫu thường được sử dụng bởi tin tặc, không được sử dụng trong các SQL được tạo ra bởi ứng dụng. Ví dụ trong MySQL nếu kẻ tấn công nghi ngờ các biện pháp an ninh phát hiện các tấn công sử dụng DROPs hoặc DELETEs, thì kẻ tấn công có thể tấn công sử dụng chuỗi có cấu trúc như sau:

```
DR/**/OP TAB/**/LE USER  
DE/**/LE/**/TE FR/**/OM USER
```

Một kỹ thuật SQL injection phổ biến liên quan đến việc sử dụng UNION ALL SELECT để lấy dữ liệu từ bất kỳ bảng nào trong hệ thống. Cú pháp cho tùy chọn SELECT là:

```
SELECT ...  
UNION [ALL | DISTINCT]  
      SELECT ..  
[UNION [ALL | DISTINCT]  
      SELECT ...]
```

UNION được sử dụng để kết hợp kết quả từ nhiều câu lệnh SELECT thành một tập kết quả. Nếu không sửa từ khóa ALL cho UNION thì tất cả các hàng trả về sẽ là duy nhất, vì vậy hầu hết các cuộc tấn công SQL injection sử dụng UNION ALL.

Tin tặc có thể sử dụng các UNIONs để có thể thêm các truy vấn bổ sung trên những truy vấn hiện có. Các thông tin hiển thị trên trang giao diện thường là kết quả truy vấn của các tìm kiếm điều kiện. Ví dụ: Giả sử rằng một trang web cho phép tìm kiếm tất cả các chuyến bay đến một thành phố nào đó. Khi thực hiện nhập vào ô nhập tên thành phố sẽ có được một danh

sách các chuyến bay. Mỗi dòng trong danh sách cho thấy các hãng hàng không, chuyến bay, thời gian khởi hành. Giả sử rằng ứng dụng này dễ bị SQL injection (tức là nó sử dụng nối chuỗi và không nhận những gì đã nhập vào trong trường thành phố được sử dụng trong mệnh đề WHERE). Thông thường câu lệnh SELECT trong ứng dụng có thể là:

```
Select airline, flightNum, departure from flights where  
city='ORD'
```

Giả sử rằng thay vì nhập ORD (chicago) vào mục tìm kiếm đầu vào, một kẻ tấn công có thể nhập chuỗi sau:

```
ORD` union all select loginame, hostname, login_time  
from master..sysprocesses  
where `1`='1'
```

Trong trường hợp này báo cáo kết quả lựa chọn sẽ là:

```
select airline, flightNum, departure from flights where  
city='ORD'  
union all select loginame, hostname, login_time from  
master..sysprocesses where '1'='1'
```

Ngoài các chuyến bay được phép liệt kê, thì bây giờ kẻ tấn công có thể xem được bất cứ người dùng nào đăng nhập vào máy chủ cơ sở dữ liệu với những tên đăng nhập của họ khi họ đăng nhập.

Airline	Flight#	Departure Date/Time
Delta	1562	8/20/2004 17:00
Delta	232	8/20/2004 19:00
JetBlue	561	8/20/2004 16:30
American	2344	8/20/2004 19:40
Continental	431	8/20/2004 22:00
British Airways	234	8/20/2004 20:00
Air Canada	114	8/20/2004 18:00
sa	fksrv	8/19/2004 01:00
sa	dbahst	8/20/2004 10:04

Hình 4.5 Thông tin đăng nhập được đính vào danh sách chuyến bay (sau một câu lệnh tiêm UNION)

Đây có thể là một khởi đầu tốt cho một cuộc tấn công. Kẻ tấn công sẽ nhận được tất cả các thông tin này trong một danh sách giống các chuyến bay như trong hình 4.5.

Một kiểu tấn công SQL injection khác là tin tặc có thể sử dụng bảng sysobject và syscolumns cho đối tượng user bằng cách sử dụng câu lệnh:

```
Select name, name, crdate from sysobjects where xtype='U'
```

Có thể nhận được một danh sách tất cả các đối tượng trong lược đồ cơ sở dữ liệu hiện tại, như hình 4.6.

Ví dụ trên được thực hiện trên SQL Server, nhưng kiểu tấn công này có thể thực hiện trên bất kỳ cơ sở dữ liệu nào. Điều duy nhất cần phải thay đổi khi áp dụng với kiểu cơ sở dữ liệu khác là thêm vào UNION SELECT – sử dụng các đối tượng thích hợp trong cơ sở dữ liệu tương ứng.

Hạn chế chính của kỹ thuật này là các cột đã lựa chọn được liệt kê ở các vị trí tương ứng của câu lệnh SELECT phải cùng loại. Trong các tài liệu giới thiệu về tấn công “SQL Injection 101”, thường chỉ ra rằng nhiệm vụ chính mà một tin tặc phải thực hiện là sử dụng UNION để tìm ra số lượng các cột và các kiểu dữ liệu mà nó phải làm việc, đây là một mốc chót để hiểu các cuộc tấn công SQL injection cơ bản sử dụng UNION. Ví dụ, nếu quan sát thấy các lỗi cơ sở dữ liệu có dạng : “all queries in SQL statements containing a

UNION operator must have an equal number of expressions in their target lists,” thì có thể thực hiện tấn công SQL injection trên cơ sở dữ liệu này.

Cuối cùng hãy xem xét một dạng SQL phổ biến – liên quan tới insert và select. Phương pháp này dựa trên một thực tế là tất cả các nhà cung cấp cơ sở dữ liệu chủ yếu đều hỗ trợ việc sử dụng các truy vấn phụ, chẳng hạn các truy vấn phụ SELECT có thể sử dụng trong một truy vấn INSERT. Ví dụ, giả sử có một màn hình cho phép nhập thông báo tới một bản tin cho nhiều người đọc, như trong hình 4.7 .

Airline	Flight#	Departure Date/Time
Delta	1562	8/20/2004 17:00
Delta	232	8/20/2004 19:00
JetBlue	561	8/20/2004 16:30
American	2344	8/20/2004 19:40
Continental	431	8/20/2004 22:00
British Airways	234	8/20/2004 20:00
Air Canada	114	8/20/2004 18:00
Orders	Orders	8/6/2000 01:34
Products	Products	8/6/2000 01:34
Order Details	Order Details	8/6/2000 01:34
CustomerCustomerDemo	CustomerCustomerDemo	8/6/2000 01:34
CustomerDemographics	CustomerDemographics	8/6/2000 01:34
Region	Region	8/6/2000 01:34
Territories	Territories	8/6/2000 01:34
EmployeeTerritories	EmployeeTerritories	8/6/2000 01:34
Employees	Employees	8/6/2000 01:34
Categories	Categories	8/6/2000 01:34
Customers	Customers	8/6/2000 01:34
Shippers	Shippers	8/6/2000 01:34
Suppliers	Suppliers	8/6/2000 01:34

Hình 4.6 Lấy được một danh sách tất cả các đối tượng người dùng bằng cách sử dụng một tấn công UNION



Hình 4.7 Thêm một tin nhắn vào một bảng tin

RECENT POSTS			
Subject	Author	Date/Time (ET)	
Re: Who is the Liberal Candidate	hale_mel_n...	11:29am, Aug 30	
Stronger dollar earnings of companies	trendingpol...	11:29am, Aug 30	
Stronger dollar earnings of companies	trendingpol...	11:29am, Aug 30	
"SECRET COURT POSSES CHALLENGES"	senilepolC...	11:29am, Aug 30	
Re: Apple using chips from IBM.	AUDIOCURE...	11:23am, Aug 30	
KELLY KIDS BOOKED @ MTV	mashable...	11:21am, Aug 30	
KELLY, DUCK AND COVERED	mashable...	11:19am, Aug 30	
Re: Apple using chips from IBM.	AUDIOCURE...	11:16am, Aug 30	
Re: Who is the Fairy Candidate	want_it...	11:14am, Aug 30	
Re: Apple using chips from IBM.	AUDIOCURE...	11:13am, Aug 30	
Who is the Liberal Candidate	harrym07...	11:04am, Aug 30	
Japan is slowing IBM. Japan big. What for	deflation...	11:01am, Aug 30	
Re: IBM & JPMC contract	nothorach...	10:57am, Aug 30	
Google money	m073_15c...	10:53am, Aug 30	
LET ME TELL YOU SOMETHING	nuffdu3hk...	10:51am, Aug 30	
IBM to lose \$50 billion JPMC contract.	newQuot09...	10:50am, Aug 30	
Re: Another IBM Market day! OIL---	want_it...	10:46am, Aug 30	
Re: Another IBM Market day! OIL---	deflation...	10:21am, Aug 30	
Re: mamma jumh	hale_mel_n...	10:18am, Aug 30	
Re: See schmutz.name	m073_15c...	10:18am, Aug 30	

Hình 4.8 Các tin nhắn trên bảng tin

Các chức năng ứng dụng có thể chèn thông báo này vào bảng MESSAGE, và cho phép tất cả các thành viên trong nhóm người đọc tin xem thông báo đó, như thể hiện trong hình 4.8.

Xây dựng một bảng thông báo là một công việc đơn giản nếu cần quan tâm đến vấn đề an ninh bảo mật (các cách mà tin tức thực hiện các tấn

công), chẳng hạn có thể dễ dàng quyết định thực hiện các chức năng bảng tin bằng việc tạo thêm một bảng MESSAGES trong cơ sở dữ liệu. Để đơn giản, giả định các cột trong bảng MESSAGES lần lượt là SUBJECT, AUTHOR, TEXT, và TIMESTAMP và timestamp được tự động tạo ra.

Trong trường hợp này, các mã ứng dụng dùng để cập nhật các thông báo được đăng vào cơ sở dữ liệu có thể viết như sau:

```
INSERT into MESSAGES(SUBJECT, AUTHOR, TEXT)
values (<whatever you type in the subject field>, <your login
name in the application>, <whatever you type in the message
text area>)
```

Chức năng đơn giản này dễ bị tấn công bởi một cuộc tấn công SQL injection đơn giản sử dụng lệnh insert và select. Nếu gõ vào sau các trường thích hợp :

```
Subject field: start`, `start`, `start`); insert into
messages(subject, author) select o.name,c.name from sysobjects
o, syscolumns c where c.id=o.id; insert into messages values
(`end Author field: end
Text field: end
```

Chuỗi SQL dưới đây sẽ được gửi tới cơ sở dữ liệu MS SQL Server.

```
INSERT into MESSAGES(SUBJECT, AUTHOR, TEXT) values
('start',
'start', 'start'); INSERT into MESSAGES (subject, author)
select o.name,c.name from sysobjects o, syscolumns c where
c.id=o.id; INSERT into MESSAGES values ('end', 'end', 'end')
```

Trong trường hợp này, ta có thể thấy tất cả tên bảng và tên cột sẽ được liệt kê trên bảng tin.

4.4.2 Quản lý, giám sát, cảnh báo và ngăn chặn.

Cơ chế tấn công SQL injection khá đơn giản và việc tấn công có thể thực hiện dưới nhiều dạng khác nhau. Chúng ta có thể chống lại mối đe dọa nghiêm trọng này bằng các biện pháp sau : (1) hạn chế các lỗ hổng của các phần mềm ứng dụng, (2) tìm ra và khắc phục các lỗ hổng SQL injection, (3) bảo vệ cơ sở dữ liệu bằng cách lọc tất cả các lệnh SQL được cấp phát bởi các phần mềm ứng dụng.

SQL injection không hẳn là một lỗ hổng của cơ sở dữ liệu. Nó có thể là lỗ hổng của các đoạn mã nguồn của phần mềm ứng dụng làm lộ cơ sở dữ liệu và dữ liệu. Tất cả các biện pháp thực hiện ở trên đều không phải là biện pháp tuyệt đối, nếu mã nguồn trong ứng dụng gây ra các lỗ hổng, thì nó sẽ không được tin cậy hoàn toàn.

Lựa chọn đầu tiên là loại bỏ các lỗ hổng ứng dụng. Đây là trách nhiệm của người phát triển ứng dụng, nhưng đôi khi nó cũng liên quan tới người quản lý cơ sở dữ liệu. Hiện nay có một số hướng dẫn tốt để tránh lỗi SQL injection cho các nhà phát triển ứng dụng, chẳng hạn như:

- Tất cả các dữ liệu được nhập vào bởi người dùng phải được lọc, loại bỏ tất cả các ký tự hoặc các chuỗi không phải là một phần của biểu thức đầu vào, và tất cả các trường nhập vào đều phải được xác nhận.
- Các lệnh SQL được sử dụng để truy nhập cơ sở dữ liệu từ mã lệnh trong phần mềm ứng dụng không được tạo ra bằng cách sử dụng các chuỗi ghép nối.
- Các tham số có kiểu mạnh (thường kết hợp với các Store procedure) phải được sử dụng bất kỳ nơi nào có thể.
- Việc đăng nhập vào ứng dụng phải sử dụng một thủ tục lưu trữ (Store procedure) được thực hiện với một thủ tục lưu trữ xác nhận tốt.

- Dữ liệu nhập vào của tất cả người dùng phải được cho vào trong dấu ngoặc kép, kể cả dữ liệu số.

Khi sử dụng các câu lệnh tiền xử lý để chống lại các chuỗi lệnh ghép nối, các chuỗi lệnh SQL được tách biệt với các chuỗi mà người dùng nhập vào. Do vậy, nó là một trong những cách đơn giản nhất để chống lại tấn công SQL injection. Với các lệnh tiền xử lý (cho hệ quản trị cơ sở dữ liệu Oracle), sẽ là:

```
update test set a = :1
```

Nếu không có lệnh tiền xử lý, nó sẽ giống như:

```
update test set a = 'ABC'
```

Bằng việc giám sát truy nhập và tạo ra báo cáo khi một ứng dụng sử dụng các lệnh tiền xử lý, chúng ta có thể hướng tới việc sử dụng các lệnh tiền xử lý rộng rãi hơn và môi trường an toàn hơn.

Tập các tham số là một tính năng hữu ích hỗ trợ việc chống dữ liệu nhập vào không như ý muốn bằng cách coi tất cả các dữ liệu đầu vào là chữ. Ví dụ, trong Microsoft SQL Server , thay vì gắn các dữ liệu nhập vào cho các chuỗi SQL của nó, chúng ta có thể sử dụng đối tượng SqlParameter như sau:

```
SqlDataAdapter command = new  
SqlDataAdapter("authenticateUser", connection);  
command.SelectCommand.CommandType =  
CommandType.StoredProcedure;  
SqlParameter parm =  
command.SelectCommand.Parameters.Add("@login",  
SqlDbType.VarChar,8); parm.Value=LoginField.Text;
```

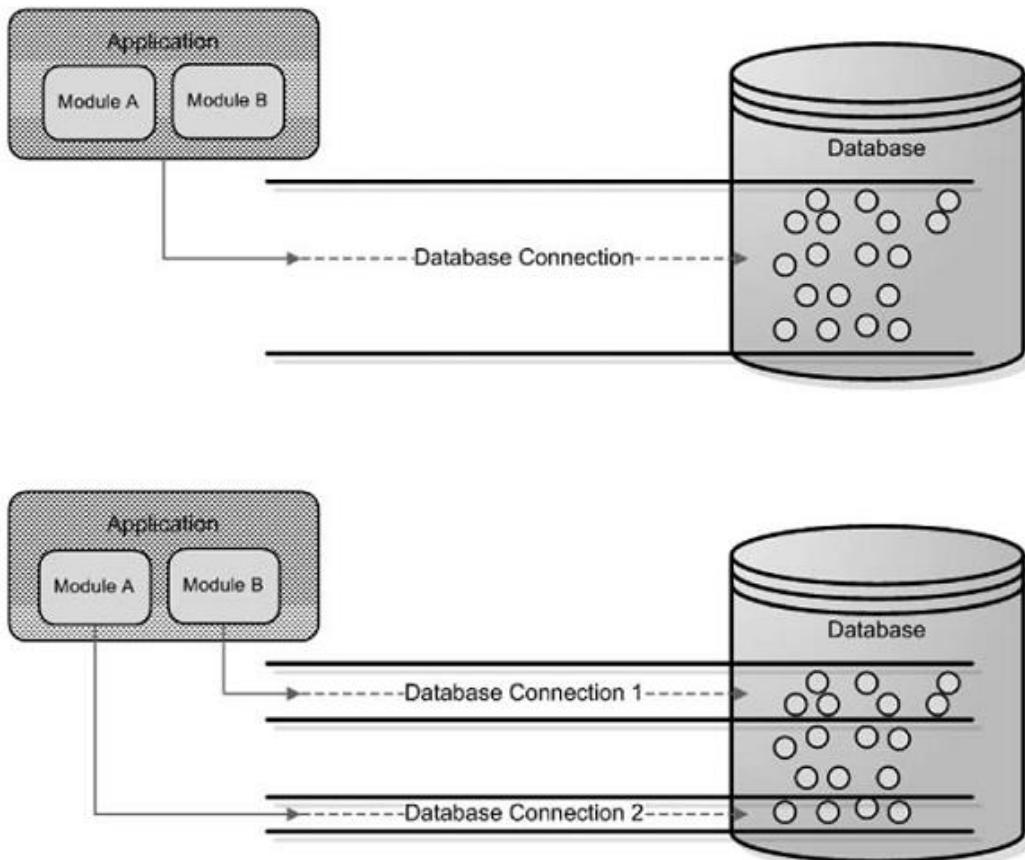
Chúng ta cũng có thể sử dụng các công cụ SQL injection để kiểm tra các phần mềm ứng dụng của mình xem có lỗi hay không. Các công cụ này thường là miễn phí. Ví dụ, SQL Injector được cung cấp như là một phần của bộ công cụ SPI của SPI Dynamics www.spidynamics.com/products/Comp_Audit/toolkit/SQLinjector.html.

Công cụ này thực hiện các cuộc tấn công SQL injection lên các phần mềm ứng dụng sử dụng Oracle hoặc Microsoft SQL Server để kiểm tra xem có lỗ hổng SQL injection hay không. Công cụ này chỉ hỗ trợ hai kiểu tấn công SQL injection cơ bản, nhưng nó vẫn hữu ích để hạn chế việc kiểm tra lỗi SQL injection một cách thủ công.

Rà soát và kiểm tra mã nguồn chỉ là một cách để chặn trước SQL injection và không phải lúc nào chúng cũng dễ dàng thực hiện. Trong nhiều trường hợp chúng ta không có thẩm quyền, hoặc được ủy quyền để chống lại các tấn công như vậy. Trong những trường hợp này, vẫn có thể làm một số giải pháp đối với các mã lệnh của phần mềm ứng dụng – tất cả đều dựa trên cơ sở thực hiện đặc quyền tối thiểu – như không cho phép các phần mềm ứng dụng đăng nhập bằng tài khoản quản trị hoặc không cho phép một ứng dụng nào đó truy nhập toàn bộ các thủ tục lưu trữ (được minh họa trong hình 4.9), từ đó có thể hạn chế những thiệt hại khi có tấn công xảy ra.

Ở đây có thể thường xuyên rơi vào tình huống mà các modul truy nhập cơ sở dữ liệu không khác nhau đối với tất cả các người dùng, chẳng hạn như truy nhập cơ sở dữ liệu được chia thành nhiều đăng nhập trong cơ sở dữ liệu, vì các nhà phát triển không muốn mạo hiểm thay đổi. Trong những trường hợp này, cách tốt nhất là ta nên tạo ra một hồ sơ truy nhập cho mỗi người dùng, và kiểm soát sự truy nhập trên hồ sơ đó. Điều này được thực hiện bằng cách ghi lại tất cả các truy nhập của người dùng và ít nhất phải lưu được các thông tin sau:

- Những cơ sở dữ liệu đang được dùng.
- Những đối tượng trong cơ sở dữ liệu (ví dụ như các bảng, thủ tục) đang được truy nhập.
- Những lệnh đang được sử dụng (ví dụ : SELECT, DML, DDL).



Hình 4.9 Áp dụng việc thực hành tốt nhất cơ chế đặc quyền tối thiểu để hạn chế kết quả hợp pháp từ những điểm yếu ứng dụng (Before – top; After – bottom)

Nên giữ lại tất cả các thông tin này trong một chu kỳ thời gian để nó phản ánh đầy đủ quy trình của mỗi ứng dụng cụ thể. Ví dụ, nếu ứng dụng có chức năng đặc biệt xảy ra vào cuối mỗi tháng thì dữ liệu ghi lại phải bao gồm tất cả các hoạt động cuối cùng của tháng. Trong hình 4.10 có thể quan sát thấy ứng dụng đã sử dụng cơ sở dữ liệu như thế nào.

```

sa      SELECT      master..sysprocesses
sa      SELECT      master.dbo.decrypttext
sa      DELETE      dbo.tblparamsettings
sa      DELETE      employees
sa      DELETE      orders
sa      DELETE      society_groups
sa      SELECT      master.dbo.syslogins
sa      UPDATE      products
sa      UPDATE      roysched
sa      UPDATE      sales
sa      UPDATE      society_groups
sa      UPDATE      t1
sa      UPDATE      titles
sa      UPDATE STATISTICS anames
sa      UPDATE STATISTICS authors
sa      UPDATE STATISTICS go
sa      UPDATE STATISTICS orderdetails
northwind SELECT      northwind..customers
northwind SELECT      northwind..Order Details
northwind SELECT      northwind..orders
northwind SELECT      northwind..products
northwind SELECT      northwind..sysmembers
northwind SELECT      northwind..sysusers
northwind SELECT      northwind.dbo.orders
northwind SELECT      northwind.dbo.products

```

Hình 4.10 Báo cáo chi tiết về truy nhập ứng dụng – ai, cái gì, như thế nào

Mặc dù có thể tạo ra chi tiết các lịch sử truy nhập dựa trên những tính năng sẵn có của cơ sở dữ liệu nhưng cũng có thể sử dụng các sản phẩm bên ngoài thay vì sử dụng các công cụ kiểm toán của chính cơ sở dữ liệu.

Nguyên nhân chính là do hiệu suất, vì có những cơ sở dữ liệu ghi lại tất cả các thông tin sẽ ảnh hưởng tới hiệu suất của cơ sở dữ liệu đó, trong khi sử dụng một công cụ bên ngoài sẽ không ảnh hưởng tới hiệu suất của cơ sở dữ liệu. Một điểm chú ý nữa là khi sử dụng cơ sở dữ liệu để tạo ra các dữ liệu kiểm toán (ví dụ trong SQL Server) của các tấn công SQL injection mà liên quan tới mọi ghi chú có các chuỗi sp_password thì có một ảnh hưởng phụ xảy ra được gọi là lẩn tránh sự kiểm toán (*audit evasion*). Nếu sử dụng một trong hàm sp_trace <...> để ghi lại các thông tin và các lệnh được thêm vào bằng cách sử dụng ký tự -- sau chuỗi sp_password thì các vết kiểm toán sẽ không được truy vấn.

Ví dụ: giả sử có một dấu vết trên các sự kiện của DBCC. Nếu chạy một lệnh TRACEON(3205), các dấu vết sẽ tạo ra một bản ghi như sau :

Audit DBCC Event

```
DBCC TRACEON(3205) SQL Query Analyzer ronb  
RON-NYHR85G9DJ\ronb  
3936  
51  
2005-02-14 01:38:37.560
```

Tuy nhiên, nếu chạy một lệnh có dạng:

```
DBCC TRACEON(3205) -- this means nothing, but let's  
say sp_password
```

Thì kết quả trả về sẽ là:

Audit DBCC Event

```
-- 'sp_password' was found in the text of this event.x`  
-- The text has been replaced with this comment for  
security reasons. SQL Query Analyzer ronb RON-  
SNYHR85G9DJ\ronb  
3936  
51  
2005-02-14 01:40:46.170
```

Khi đã có hồ sơ sử dụng, thì có thể kiểm tra được truy nhập bằng ứng dụng có sử dụng đúng quyền mà nó được phép không. Việc này sẽ cho phép thực hiện giám sát, phát hiện và loại bỏ được nhiều tấn công SQL Injection tiềm năng. Cũng có ý kiến cho rằng điều này là không cần thiết vì theo thời gian các phản ứng đó là quá muộn và các tin tức đã lấy được tất cả cơ sở dữ liệu. Tuy nhiên các cuộc tấn công thường mất nhiều thời gian. Trừ trường hợp môi trường ứng dụng bị phá vỡ và mã hóa kém, nếu không các tin tức phải mất rất nhiều thời gian để thử nghiệm và tìm ra phương pháp đúng cho SQL

injection để lấy được dữ liệu mong muốn. Nếu có một giải pháp giám sát tốt được thiết lập giống như thông báo theo thời gian thực và một cơ sở hạ tầng để ứng phó sự cố tốt thì có thể ngăn chặn một cuộc tấn công khi nó đang diễn ra và thậm chí có thể tìm được thông tin của tin tặc, như địa chỉ IP... Lý do thứ hai là nếu xác định được một cuộc tấn công SQL injection, ta có thể loại bỏ các lỗ hổng trong ứng dụng và nâng cao an toàn theo thời gian.

Để phát hiện ra các tấn công cần phải dựa vào ba tham số chính sau: Các dấu hiệu tấn công, các ngoại lệ (các lỗi SQL) hoặc độ lệch so với hồ sơ đã được định nghĩa.

Cách đơn giản nhất để thực hiện việc theo dõi các dấu hiệu tấn công là sử dụng các hệ thống phát hiện xâm nhập trái phép (IDS) hoặc là các hệ thống tương tự IDS. Ý tưởng ở đây là xác định các mẫu nhất định (gọi là dấu hiệu của các cuộc tấn công) và tìm kiếm chúng. Dấu hiệu sẽ tương ứng với các kỹ thuật thường được sử dụng trong SQL injection. Ví dụ, có thể tìm kiếm dấu hiệu như `1=1` hoặc `UNION SELECT` hay mệnh đề `WHERE` xuất hiện sau một dấu ghi chú `(--)`. Một vấn đề với phương pháp này là có quá nhiều phương pháp để thực hiện các cuộc tấn công SQL injection và dấu hiệu tấn công đó có thể khớp với điều gì đó được coi là hợp pháp. Để minh họa cho vấn đề này, đầu tiên có thể nghĩ ra rất nhiều phép toán mà giá trị luôn đúng, nó có thể là `'1'='1` `'a'='a'` hoặc `'my dog'='my dog'`.... Thực sự là có vô số cách khác nhau, điều này cũng đúng khi tránh kiểm toán bằng mẫu UNION SELECT. Vấn đề thứ hai là những dấu hiệu có thể được sử dụng trong các hệ thống ứng dụng, ví dụ không phải chỉ có các tấn công mới sử dụng UNION ALL, và đây là lý do tại sao SQL lại hỗ trợ chức năng này. Vì vậy IDS có thể cảnh báo về một hành vi hoàn toàn hợp pháp – đó là một phát hiện sai lầm loại một(false - positive).

Cách thứ hai để phát hiện tấn công là liên quan đến lỗi SQL, các cuộc tấn công SQL injection hầu như luôn liên quan đến lỗi SQL. Xem lại các ví dụ về UNION SELECT ở phần trước (kết quả thể hiện trong hình 4.5 và 4.6), vấn đề gì sẽ xảy ra nếu tin tặc đưa vào chuỗi SQL theo mẫu : `select name, name, crdate from sysobjects where xtype='U'`

Nếu kẻ tấn công đầu tiên sử dụng chuỗi :

```
select name, crdate from sysobjects where xtype='U'
```

Các lỗi sau sẽ được trả về từ các cơ sở dữ liệu khác nhau :

SQL Server:

```
Server: Msg 205, Level 16, State 1, Line 1
```

Tất cả các truy vấn trong một câu lệnh SQL chứa phép UNION phải có một số lượng bằng nhau các biểu thức trong danh sách đích..

Oracle:

```
ORA-01789: query block has incorrect number of  
result columns
```

Sybase:

```
Msg 205, Level 16, State 1:  
Server '---', Line 2:
```

Tất cả các truy vấn trong một lệnh SQL chứa hoạt động set phải có một số lượng bằng nhau các biểu thức trong danh sách đích.

DB2:

```
DBA2191E SQL execution error.  
A database manager error occurred. :  
[IBM][CLI Driver][DB2/NT] SQL0421N The  
operands of a set operator or a VALUES clause do not  
have the same number of columns. SQLSTATE=42826
```

MySQL:

```
ERROR 1222: The used SELECT statements have a  
different number of columns
```

Như ta đã thấy các cơ sở dữ liệu luôn trả về một lỗi. Nếu theo dõi chặt chẽ các lỗi trả về từ cơ sở dữ liệu chúng ta có thể xác định được các tấn công SQL injection. Một số lỗi cần biết khi xác định SQL injection :

- Lỗi trên số cột trong lệnh SELECT (thường là UNION).
- Lỗi gây ra bởi dấu ngoặc kép.
- Lỗi gây ra bởi việc chuyển đổi dữ liệu.

Bởi vì kẻ tấn công muốn xem kết quả của các chuỗi SQL được đưa vào và họ có thể chỉnh lại các chuỗi SQL cho phù hợp, đôi khi họ còn sử dụng các kỹ thuật mà chúng ta đã biết như trong [2]. Kỹ thuật này được dựa trên việc mở một kết nối từ cơ sở dữ liệu bị tấn công tới một nguồn cung cấp dữ liệu khác thường trên máy tính của kẻ tấn công hoặc một máy chủ mà đã bị kẻ tấn công xâm nhập. Bởi vì tất cả các cơ sở dữ liệu này đều không bị “cô lập”, tất cả chúng đều có thể kết nối đến một cơ sở dữ liệu từ xa. Nếu tin tặc có thể tạo ra một kết nối và kết quả của truy vấn SQL được gửi tới một vị trí mà nó có thể nhận được kết quả đó thì có thể vượt qua tầng che giấu lỗi của cơ sở dữ liệu bị tấn công.

Một ví dụ về kỹ thuật này trong ngữ cảnh của Microsoft SQL Server là việc sử dụng OPENROWSET và OPENDATASOURCE của nhà cung cấp OLEDB. Ví dụ, kẻ tấn công muốn lấy thông tin của sysobject trên máy chạy SQL Server trên địa chỉ IP 192.168.1.168, giả sử cổng 80 để không bị chặn bởi tường lửa, thì có thể thực hiện các cuộc tấn công của bằng cách đưa vào các chuỗi SQL sau đây thông qua các ứng dụng của SQL Server chạy trên máy tính của kẻ tấn công:

```
SELECT * FROM OPENROWSET ('SQLoledb',
    uid=sa;pwd=mypwd;network=DBMSSOCN;address=192.168.1
    .168,80;', 'SELECT * FROM copied_sysobjects')
SELECT * FROM master..sysobjects
```

Trong trường hợp này, nội dung của sysobject trên cơ sở dữ liệu bị tấn công sẽ được gửi tới máy tính của của kẻ tấn công. Những lệnh này cũng cần được theo dõi như là dấu hiệu của một cuộc tấn công.

Phương pháp thứ ba và cuối cùng để xác định (và ngăn chặn) các cuộc tấn công SQL injection là việc sử dụng một cơ chế để xác định “những điều tồi tệ có thể xảy ra”. Thay vì sử dụng những dấu hiệu để phát hiện nó, chúng ta có thể theo dõi và ghi lại các hoạt động của ứng dụng trong những lần hoạt động bình thường. Cách tốt nhất để nhận được cảnh báo khi bị tấn công SQL injection đó là kiểm tra lại trên cơ sở ban đầu và nếu các truy vấn SQL không hợp lệ sẽ có một cảnh báo được tạo ra.

Có thể diễn đạt như một chính sách, tương tự như một chính sách được xác định trong tường lửa của SQL. Chính sách này sẽ bao gồm hai quy tắc. Quy tắc đầu tiên là sẽ cho phép tất cả các truy vấn SQL trên cơ sở là những câu lệnh hợp lệ và quy tắc thứ hai sẽ cảnh báo mọi truy vấn SQL còn lại. Vì các luật được áp dụng theo thứ tự từ trên xuống, nên bất kỳ yêu cầu nào gửi đến sẽ được xét bởi quy tắc đầu tiên. Nếu nó khớp với luật đầu tiên, nó sẽ được cho phép. Và ngược lại, nó sẽ bị đánh giá bởi quy tắc thứ hai. Bằng cách thay đổi hành động của quy tắc thứ hai từ ALERT sang REJECT, không chỉ giúp cảnh báo SQL injection mà còn ngăn chặn SQL injection và bảo vệ cơ sở dữ liệu an toàn. Hai chính sách (với hai quy tắc) được thể hiện trong hình 4.11.

The screenshot displays two separate policy configuration windows, each titled "Policy".

Policy Description: Alert SQL Injection

Select	Sequence	Rule Description	Rule Type	Client IP	Server IP	Source Program	DB User Group	Application User	Object	Command	Period	Exception Type	Minimum Count	Reset Interval	Continue Log Action	
<input checked="" type="checkbox"/>	1	BASELINE											-	0	0	<input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/>
<input checked="" type="checkbox"/>	2	Alert all	Access Rule	ANY	ANY	ANY		ANY	ANY	ANY	ANY		-	0	0	<input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/>

Policy Description: Block SQL Injection

Select	Sequence	Rule Description	Rule Type	Client IP	Server IP	Source Program	DB User Group	Application User	Object	Command	Period	Exception Type	Minimum Count	Reset Interval	Continue Log Action	
<input checked="" type="checkbox"/>	1	BASELINE											-	0	0	<input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/>
<input checked="" type="checkbox"/>	2	Reject all	Access Rule	ANY	ANY	ANY		ANY	ANY	ANY	ANY		-	0	0	<input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/>

Buttons at the bottom of both windows include: Select All, Unselect All, Remove, Bulk Modify, Add Access Rule..., Add Exception Rule..., Suggest Rules, and Suggest from DB AC.

Hình 4.11 Các chính sách để cảnh báo và chặn các tấn công SQL injection

Trước khi chuyển sang phần tiếp theo, có một chú ý nhỏ về các ví dụ trước. Nhiều ví dụ về SQL injection ở trên sử dụng Microsoft SQL Server. Tuy nhiên, các cơ sở dữ liệu khác nhau có thể bị tấn công nhiều hơn hoặc ít hơn tùy từng trường hợp. Nguyên nhân là do có nhiều chức năng thuận tiện hơn cung cấp bởi SQL Server, và các chức năng đó có thể bị lợi dụng bởi tin tặc :

- SQL Server hỗ trợ nhiều truy vấn nối bằng dấu chấm phẩy (;), cho phép thêm vào một truy vấn bổ sung trong những phần mềm ứng dụng thường được sử dụng.
- SQL Server hỗ trợ ký tự chuyển (--) làm đoạn sau dấu này chuyển thành phần ghi chú (phần sau sẽ bị bỏ qua, không được coi là một phần của câu truy vấn).
- SQL Server hỗ trợ chuyển đổi ngầm các chuỗi, làm cho cuộc tấn công UNION SELECT trở nên dễ dàng hơn.
- SQL Server có các thông tin về thông báo lỗi, nó rất tuyệt vời cho các nhà phát triển và DBA nhưng cũng rất tốt cho các tin tặc.

4.5 CẢNH GIÁC TRƯỚC SỰ KẾT HỢP GIỮA LỖ HỒNG SQL INJECTION VÀ TRÀN BỘ ĐỆM

SQL injection là một loại hình tấn công phổ biến có thể cho phép một tin tặc chiếm đặc quyền cao nhất (root) của hệ điều hành. Ở đây, không giới thiệu thêm bất kỳ kiến thức mới nào về tấn công, mà chỉ ra cách kết hợp các vấn đề mà chúng ta đã biết – trong trường hợp này là lỗ hổng tràn bộ đệm và SQL injection.

Phân tích lỗ hổng: Đưa các chuỗi dài vào trong các thủ tục với lỗ hổng tràn bộ đệm

Hầu hết, các cuộc tấn công SQL injection trong thực tế xuất phát từ các ứng dụng sử dụng việc ghép chuỗi để thực hiện các lệnh SQL mà các nhà

phát triển ứng dụng không mong muốn. Trong phần này, chúng ta có thể thấy một cuộc tấn công SQL injection sẽ xảy ra ngay cả khi không xuất hiện việc ghép chuỗi. Cuộc tấn công này có thể xảy ra bất cứ lúc nào khi trong cơ sở dữ liệu chứa những thủ tục có lỗ hổng tràn bộ đệm, và các tham số truyền vào thủ tục đó có thể được đưa vào bởi người dùng ứng dụng bình thường. Đây là kỹ thuật phổ biến, và có thể được sử dụng trong mọi hệ cơ sở dữ liệu, nhưng để cụ thể hơn, chúng ta sẽ xem xét hệ quản trị cơ sở dữ liệu Oracle. Vấn đề này đã được đưa ra trong một hội thảo bảo mật từ tháng 2 năm 2004 bởi Integrity, và Oracle đã phát hành bản vá để giải quyết những vấn đề này. Tại thời điểm đó, Oracle 8i và 9i bao gồm 6 hàm chức năng có lỗ hổng tràn bộ đệm. Các hàm này là một phần lõi của hệ quản trị cơ sở dữ liệu và nó không thể bị giới hạn chuỗi nhập vào :

- ⑩ BFILENAME—Oracle 8i, 9i
 - ⑩ FROM_TZ—Oracle 9i
 - ⑩ NUMTODSINTERVAL—Oracle 8i, 9i
 - ⑩ NUMTOYMINTERVAL—Oracle 8i, 9i
 - ⑩ TO_TIMESTAMP_TZ—Oracle 9i
 - ⑩ TZ_OFFSET—Oracle 9i

Ví dụ, FROM_TZ là một hàm chuyển đổi giá trị của nhãn thời gian và múi giờ sang một nhãn thời gian với giá trị múi giờ. Giá trị múi giờ đưa vào được xem như một chuỗi ký tự có định dạng giờ: phút. Chẳng hạn, nếu muốn lấy thời gian hiện tại của múi giờ Eastern, chúng ta có thể thực hiện câu lệnh select sau:

```
SELECT FROM_TZ(TIMESTAMP '2004-09-07 18:00:00', '5:00')  
FROM DUAL;
```

Và hàm FROM_TZ sẽ bị lợi dụng để tấn công khi nhập vào một chuỗi dài trong tham số múi giờ. Nếu lệnh sau được cấp quyền, lỗi tràn bộ đệm sẽ xảy ra:

Và nếu đã xây dựng được chuỗi dài theo mục đích của mình, chúng ta có thể kiểm soát được địa chỉ mong muốn trên ngăn xếp. Bởi vì Oracle chạy trên windows với tài khoản người quản trị cao nhất (administrator) nên ta có toàn quyền trên máy chủ đó. Trong UNIX, Oracle chỉ chạy với tài khoản người dùng Oracle bình thường), nên các quyền đối với cơ sở dữ liệu sẽ bị giới hạn.

Nói về SQL injection, giả sử một người dùng được yêu cầu nhập cả thời gian và múi giờ cho một giao dịch kinh doanh nào đó, và chức năng FROM_TZ được sử dụng để ghi lại thời gian dựa trên múi giờ được nhập vào. Nếu ứng dụng không kiểm tra các trường nhập vào và chấp nhận tất cả các chuỗi nhập vào bởi người sử dụng như một tham số gọi tới chức năng khác, thì hệ thống này đã có một lỗ hổng nghiêm trọng.

4.6 LỚP ỨNG DỤNG TRÊN MÁY CHỦ

Sau khi xem các vấn đề xảy ra ở lớp ứng dụng, có thể viết và triển khai các mã ứng dụng trực tiếp trong các máy chủ cơ sở dữ liệu sử dụng gói và phần mềm mở rộng được cung cấp bởi các nhà cung cấp cơ sở dữ liệu. Một số chuyên gia có thể cố gắng thuyết phục rằng điều này sẽ làm đơn giản hóa môi trường và tăng tính bảo mật. Nhưng không nên làm điều này (Ví dụ: Đó là một ứng dụng tùy chỉnh và nó hoàn toàn có thể gói gọn trong các máy chủ cơ sở dữ liệu), vì rất có thể điều này sẽ làm cho tình hình tồi tệ hơn.

Nếu chạy tất cả mọi thứ trong cơ sở dữ liệu sẽ không đưa ra các lỗi ứng dụng, những lỗi này sẽ được chạy trực tiếp trong cơ sở dữ liệu. Ngoài ra, không phải lo lắng về nhiều vấn đề chẳng hạn như: cross-site scripting, cookie poisoning, session hijacking, ... Nếu đang chạy tất cả mọi thứ trên một máy chủ, kẻ tấn công có thể tìm thấy một lỗ hổng nào đó và có thể “mở rộng” các lỗ hổng này bằng cách sử dụng cơ sở dữ liệu hoặc máy chủ web hay bất kỳ thành phần nào khác.

Ví dụ: Một kẻ tấn công có thể sử dụng SQL injection để gọi một thủ tục bên ngoài rồi sửa đổi file cấu hình trên máy chủ Web hoặc máy chủ ứng dụng, do đó hoàn toàn có thể mở ra hệ thống để truy nhập Web. Ta có thể sử dụng nhiều sản phẩm bảo mật và áp dụng các chiến lược bảo vệ theo chiều sâu cùng công cụ bảo mật như tường lửa ứng dụng, tường lửa cơ sở dữ liệu và hệ thống phát hiện xâm nhập cơ sở dữ liệu để bảo vệ môi trường của hệ thống.

Ở đây, chỉ muốn cảnh báo rằng tất cả các ứng dụng chỉ nên đăng nhập vào cơ sở dữ liệu trong điều kiện các lỗ hổng ứng dụng được xem xét và chắc chắn rằng không có sai sót nào. Hơn nữa các máy chủ cơ sở dữ liệu phức tạp hơn (với những kiểu chức năng nó hỗ trợ) thì các lỗi sẽ nhiều hơn và các lỗ hổng có thể bị khai thác nhiều hơn.

Ví dụ: Nếu máy chủ có thể xử lý các dịch vụ Web, khi đó nhiều mã được thực thi như một phần của máy chủ. Tuy nhiên, càng nhiều mã sẽ có nhiều lỗi hơn, do đó có thể có nhiều cách để tấn công vào cơ sở dữ liệu hơn.

4.7 BỘ ĐÓNG GÓI ỨNG DỤNG

Khi nói đến lỗ hổng ứng dụng, chúng ta sẽ thắc mắc rằng những lỗ hổng này ảnh hưởng đến cơ sở dữ liệu như thế nào. Có hai vấn đề cần quan tâm, đó là:

- 1: Chúng ta hiểu biết nhiều hơn về các ứng dụng của riêng mình so với bộ đóng gói.
- 2: Các nhà phát triển ứng dụng trong tổ chức thường có những thói quen xấu.

Quan điểm này có phần hợp lý nhưng không hoàn toàn như vậy. Mặc dù đóng gói bộ ứng dụng được nhiều người có kinh nghiệm lâu năm phát triển và thử nghiệm (thường là thử nghiệm để tốt hơn và an toàn hơn), nhưng vẫn có rất nhiều lỗ hổng bảo mật. Trong thực tế những bộ phần mềm của các công ty như SAP, Oracle, PeopleSoft và Siebel với các hàm có kích thước lớn, chúng chắc chắn sẽ có lỗi. Trong số các gói này có hàng triệu dòng code, thường

được viết bằng nhiều ngôn ngữ lập trình của nhiều thể hệ các nhà phát triển. Hơn nữa, bởi vì các hệ thống này được sử dụng cho các doanh nghiệp, cho nên họ thường thiết kế và tùy biến ngoài phần lõi của sản phẩm, các sản phẩm tùy chỉnh thường được triển khai trong thời gian ngắn và thậm chí còn ít thử nghiệm. Nếu trong một doanh nghiệp lớn có một trong các bộ ứng dụng được cài đặt, lý do là các hệ thống này được sử dụng cho Enterprise Resource Planning (ERP - Hoạch định nguồn lực doanh nghiệp), Customer Relationship Management (CRM - Quản lý quan hệ khách hàng), Supply Chain Management (SCM - Quản lý chuỗi cung ứng). Như thế những bộ ứng dụng thường có một kết nối trực tiếp vào cơ sở dữ liệu. Là chủ sở hữu của cơ sở dữ liệu, ta cần phải tìm hiểu về những loại lỗ hổng bảo mật của các ứng dụng này.

Phân tích các lỗ hổng: Tất cả các lỗi ứng dụng

Trong quá trình sửa lỗi, có thể loại bỏ được một số lỗi. Những bộ phần mềm ứng dụng lớn thường có các lỗ hổng bảo mật riêng của nó.

Ví dụ: Oracle E-Business suite phiên bản 11.0.x và phiên bản 11.4.1 đến phiên bản 11.4.8 có nhiều lỗ hổng SQL injection cho phép một kẻ tấn công SQL injection vào các trường đầu vào trên các mẫu Web. Do thiết kế và độ tin cậy giữa một cơ sở dữ liệu Oracle và ứng dụng, các cuộc tấn công có thể làm ảnh hưởng tới toàn bộ cơ sở dữ liệu. Một vài cảnh báo về bảo mật và các phiên bản ứng dụng Oracle mà họ nói đến được thể hiện trong bảng 4.1

Số cảnh báo bảo mật của Oracle	Phiên bản ứng dụng Oracle có lỗ hổng
32	11.4.1-11.4.6
44	11i
53	10.7-11.4.8
56	11.4.1-11.4.8
57	11.0.x, 11.4.1-11.4.8

Bảng 4.1 Các cảnh báo bảo mật cho ứng dụng Oracle

Tiếp tục với các ví dụ về ứng dụng của Oracle và cơ sở dữ liệu Oracle. Những vấn đề khác mà chúng ta gặp phải ngoài SQL injection là gì? Lỗ hổng tồn tại trong ứng dụng Oracle – có khoảng 15 tài khoản mặc định, mật khẩu

mặc định và các cấu hình mặc định phải được thay đổi hoặc là loại bỏ. Theo mặc định không giới hạn số lần đăng nhập thất bại do vậy ở đây vấn đề crack mật khẩu cũng là một lỗ hổng. Các ứng dụng Oracle truy nhập vào cơ sở dữ liệu bằng cách sử dụng tài khoản APPS, tuy nhiên không có thông tin được truyền qua cơ sở dữ liệu để kiểm soát chặt chẽ quá trình truy nhập và thực hiện dữ liệu.

Ví dụ, trong một triển khai đầy đủ các ứng dụng Oracle, thường kèm theo các cổng được quan tâm như bảng 4.2:

Máy chủ	Cổng
Máy chủ cơ sở dữ liệu Oracle	1521
Máy chủ ứng dụng Oracle	80 , 443 và đôi khi 2649, 8888 và 7777
Biểu mẫu Forms Listener Oracle	9000
Oracle WebDB Listener	2002
Máy chủ TCF Oracle	10.021-10.029 , 15000
Oracle Report Review Agent	1526
Máy chủ Metric Oracle	9010 , 9020

Bảng 4.2 Cổng cho máy chủ ứng dụng Oracle

4.8 HƯỚNG TÓI LIÊN KẾT GIỮA MÔ HÌNH NGƯỜI SỬ DỤNG VÀ MÔ HÌNH CƠ SỞ DỮ LIỆU NGƯỜI DÙNG

Cơ sở dữ liệu là mô hình bảo mật toàn vẹn và chúng ta luôn luôn cố gắng sử dụng nó tới mức tối đa có thể. Mô hình này dựa trên các quyền kết hợp với một đăng nhập cơ sở dữ liệu liên quan tới thông tin đăng nhập cơ sở dữ liệu, xác thực, ủy quyền. Một trong những vấn đề liên quan đến bảo mật cơ sở dữ liệu trong kiến trúc ba tầng và các ứng dụng Web là mô hình người sử dụng ứng dụng với mô hình cơ sở dữ liệu người dùng. Cần phân biệt giữa người dùng của ứng dụng và người dùng của cơ sở dữ liệu. Người sử dụng ứng dụng thường không có ánh xạ trực tiếp đến thông tin đăng nhập trong cơ sở dữ liệu có nghĩa là đặc quyền cơ sở dữ liệu không thể sử dụng để hạn chế truy nhập vào dữ liệu hoặc hoạt động. Điều này là không an toàn vì nó có nghĩa là mỗ

hình bảo mật cơ sở dữ liệu không thể được sử dụng để hạn chế và kiểm soát những kết nối ứng dụng.

Để tránh điều này, nên hướng tới sự kết hợp người dùng của hai mô hình này. Điều này sẽ cho phép chúng ta thực thi cấp bảo mật người dùng trong cơ sở dữ liệu, không nhất thiết phải là một sự thay thế cho mô hình bảo mật ứng dụng, mà như một cơ chế hỗ trợ cho việc kiểm soát người dùng mức ứng dụng.

4.9 TÓM TẮT

Chương này đã đề cập đến vấn đề bảo mật cơ sở dữ liệu với ứng dụng. Cụ thể hơn, chương này giúp bạn đọc hiểu rõ về một số đặc điểm của các ứng dụng, một số trong đó có thể giúp bạn đọc trong việc tạo ra một môi trường cơ sở dữ liệu an toàn (chẳng hạn như xác định tính chất của các lệnh gọi SQL được tạo ra bởi các ứng dụng).

Điều quan trọng nhất trong chương này là ngay cả khi vấn đề không phải là một phần của lớp cơ sở dữ liệu thì chúng ta cũng cần quan tâm đầy đủ các khía cạnh khác nhau để đảm bảo cho an toàn cơ sở dữ liệu tổng thể. Một chủ đề đã được đề cập ngắn gọn trong cuối chương này là vấn đề cải thiện bảo mật tổng thể bằng việc kết hợp giữa các mô hình ứng dụng bảo mật và mô hình bảo mật cơ sở dữ liệu.

4.10 CÂU HỎI ÔN TẬP

1. Nêu một số hiểm họa có thể xảy ra ở mức ứng dụng
2. Vấn đề quản lý tài khoản và mật khẩu người dùng ở mức ứng dụng cần xem xét những vấn đề gì?
3. Phân tích một số điểm yếu tại mã nguồn ứng dụng
4. Mô tả các phương thức tấn công của SQL Injection và cách phòng tránh.

CHƯƠNG 5. AN TOÀN TRONG CƠ SỞ DỮ LIỆU THÔNG KÊ

Trong chương này đi sâu vào các vấn đề suy diễn trên các SDB, đặc biệt quan tâm đến các kỹ thuật bảo vệ có sẵn. Trong chương trình bày các kỹ thuật bảo vệ cơ bản. Các kỹ thuật này dựa vào việc hạn chế các thông kê mà người dùng yêu cầu, đây là các thông kê cho phép người sử dụng suy diễn thông tin bí mật. Các kỹ thuật bảo vệ này cũng dựa vào việc xáo trộn dữ liệu, đây là các dữ liệu được sử dụng cho việc tính toán các thông kê yêu cầu. Cuối cùng là đánh giá chung về đặc trưng của các kỹ thuật này.

5.1 GIỚI THIỆU

Cơ sở dữ liệu thống kê là một loại cơ sở dữ liệu chứa những thông tin cơ bản nhưng không cho phép các câu truy vấn tiết lộ dữ liệu riêng tư của các đối tượng hay cá nhân trong cơ sở dữ liệu. Lưu ý rằng cá nhân ở đây có thể là người hoặc vật. Loại cơ sở dữ liệu này là một giải pháp cho vấn đề bí mật trong hầu hết các hệ thống lưu trữ thông tin mật (như cơ sở dữ liệu nhân khẩu và các bản ghi tín dụng) là các hệ thống chứa thông tin cần chia sẻ nhưng phải hạn chế truy nhập vào các thông tin riêng của mỗi cá nhân hay tổ chức.

Tuy nhiên một điểm yếu của loại cơ sở dữ liệu này là vấn đề *suy diễn*: người dùng có thể suy ra các thông tin bí mật từ những truy vấn thông kê hợp lệ. Đây là một thách thức đối với hệ thống này. Các biện pháp đảm bảo an toàn cho cơ sở dữ liệu thống kê có thể áp dụng với các truy vấn thực hay bắn thân dữ liệu, điều đó tùy thuộc vào mức độ an toàn mà chúng ta muốn đạt được.

Có thể định nghĩa một *cơ sở dữ liệu thống kê* (*SDB*) là một cơ sở dữ liệu chứa các bản ghi mô tả về các cá nhân nhưng chỉ các thông tin thống kê mới sẵn dùng, có nghĩa là chỉ các câu truy vấn thống kê (như: COUNT, SUM, MEAN, MAX, MIN...) mới được trả lời, ngoài các câu truy vấn này thì những truy vấn vào các mục dữ liệu riêng sẽ không được đáp lại. Ban đầu các

SDB chỉ được sử dụng cho các tính toán thống kê (chẳng hạn như SDB điều tra dân số). Sau đó, chúng được sử dụng cho các cơ sở dữ liệu thông thường (ví dụ như các SDB của bệnh viện, ngân hàng, học viện, v.v).

Các SDB được dùng trong nhiều ứng dụng, như cơ sở dữ liệu điều tra dân số, cơ sở dữ liệu về số người tử vong, về kế hoạch kinh tế, cơ sở dữ liệu thống kê về khám chữa bệnh, cơ sở dữ liệu về các vụ tai nạn ô tô, cơ sở dữ liệu về công nhân...

5.1.1 Dạng biểu diễn của cơ sở dữ liệu thống kê

Một cơ sở dữ liệu thống kê (SDB) khác với các cơ sở dữ liệu bình thường ở chỗ là khả năng truy vấn của nó bị giới hạn. Việc truy vấn chỉ được giới hạn ở một vài phép toán thống kê như: đếm, tính tổng, tính giá trị trung bình, và một vài phép toán thống kê khác. Các lược đồ dữ liệu của SDB không cần thiết phải khác với các lược đồ dữ liệu của các cơ sở dữ liệu khác. Các dữ liệu trong SDB đều được hình thức hóa bằng một lược đồ quan hệ.

Có hai dạng biểu diễn của cơ sở dữ liệu thống kê là: dạng quan hệ, dạng vĩ mô. Để đơn giản hóa, một SDB có thể được mô hình hóa bằng một quan hệ R đơn. Còn trong thực tế, hầu hết các SDB đều được thực thi bằng các cơ sở dữ liệu lớn với nhiều quan hệ.

Nói chung, các SDB có thể được biểu diễn dưới dạng bảng hai chiều bình thường như các cơ sở dữ liệu quan hệ khác. Tuy nhiên với các SDB có mục đích đặc biệt (như SDB điều tra dân số chặng hạn) thì SDB có thể được biểu diễn bằng các bảng chứa các thống kê vĩ mô hay còn gọi là dạng vĩ mô. Bởi vì, trong một số trường hợp, các kỹ thuật kiểm soát suy diễn chỉ được phát triển trên các SDB mà đưa ra các thống kê dưới dạng bảng vĩ mô.

Dưới đây là ví dụ về dạng quan hệ và dạng vĩ mô của cơ sở dữ liệu thống kê về công nhân của một công ty.

Dạng quan hệ:

ID	Ten	ChucVu	Phong	Tuoi	GioiTinh	Lương
01	Phương	Nhân viên	Maketing	30	F	3700
02	Tuyến	Trưởng phong	Kế hoạch	42	M	6500
03	Phương	Nhân viên	Kế hoạch	19	M	3800
04	Huyền	Giám sát viên	Maketing	28	F	4500
05	Phương	Nhân viên	Maketing	25	M	2500
06	Thắng	Phó phòng	Tài vụ	39	M	3900

Bảng 5.1 Ví dụ về SDB dạng quan hệ

Dạng vĩ mô (với thống kê Count)

BSD Table				
Birth-Year	Sex	Dept-Code		
		Dept1	Dept2	Dept3
1941-1951	M	10	12	0
	F	1	0	3
1952-1962	M	12	10	5
	F	20	2	8
>1962	M	15	0	1
	F	20	10	0

Bảng 5.2 Ví dụ về thống kê vĩ mô về công nhân

5.1.2 Các cơ sở dữ liệu thống kê trong thực tế

Trong thực tế, cơ sở dữ liệu thống kê có rất nhiều ứng dụng trên thế giới. Các quốc gia đều sử dụng cơ sở dữ liệu thống kê cho nhiều mục đích riêng. Mỗi quốc gia hay tổ chức đều có website riêng dành cho các thống kê về nhiều lĩnh vực. Những tổ chức, cá nhân có nhu cầu có thể xem những thống kê khác nhau trên các website này.

Chẳng hạn, một số cơ sở dữ liệu thống kê của các tổ chức và quốc gia trên thế giới được kể tên dưới đây:

Cơ sở dữ liệu thống kê Liên hợp quốc nhằm sử dụng để:

- + Thống kê thương mại hàng hoá
- + Thống kê dân số, nhà ở
- + Thống kê số người tử vong
- + vv

Cơ sở dữ liệu thống kê kinh tế khối Châu Âu nhằm một số mục đích như:

- + Thống kê kinh tế
- + Thống kê lâm nghiệp
- + Thống kê giới tính
- + Thống kê vận tải
- + vv

Trang web của Tổ chức Thương mại thế giới (WTO) đã xây dựng cơ sở dữ liệu thống kê theo thời gian (Time Series on international trade), người dùng tùy theo nhu cầu, có thể truy nhập vào trang web <http://wto.org> để tra cứu cơ sở dữ liệu về thương mại hàng hóa và dịch vụ quốc tế.

Các dữ liệu được trình bày theo sản phẩm và nhóm khu vực. Chia thành 6 bộ dữ liệu: các mặt hàng (Merchandise trade by commodity), các chỉ số thương mại (Merchandise trade indices), mạng lưới thương mại thế giới (Network of world merchandise trade), các hiệp định hội nhập được chọn (Selected regional integration agreements), tổng thương mại hàng hóa (Total merchandise trade), các dịch vụ thương mại (Trade in commercial services).

Cơ sở dữ liệu thống kê của tổ chức thương mại thế giới (WTO) bao gồm:

- + *Hồ sơ thành viên Thương mại (Trade Profiles)*: cung cấp các thông tin được xác định trước về tình hình thương mại của các thành viên
- + Các *Hồ sơ thuế quan (Tariff Profiles)*: cung cấp thông tin về tình hình tiếp cận thị trường của các thành viên.
- + *Hồ sơ thành viên Dịch vụ (Services Profiles)*: cung cấp các số liệu thống kê chi tiết về các dịch vụ cơ sở hạ tầng trọng điểm (giao thông vận tải, viễn thông, tài chính, bảo hiểm) đối với nền kinh tế được lựa chọn.

Cơ sở dữ liệu thống kê của Thụy Điển bao gồm:

- + Thống kê nông nghiệp, lâm nghiệp, thuỷ sản
- + Thống kê ngành nghề kinh doanh
- + Giáo dục và nghiên cứu
- + Môi trường
- + Thị trường tài chính
- + Dân số
- + Giá cả và tiêu dùng
- + Tài chính công
- + Thương mại hàng hoá và dịch vụ

Để tham khảo cách tra cứu số liệu thống kê chuỗi thời gian trên trang web của WTO, bạn đọc có thể xem phần phụ lục.



Statistics database

Welcome to the WTO statistics database, which allows you to retrieve statistical information in the following presentations.

- The *Trade Profiles* provide predefined information leaflets on the trade situation of members, observers and other selected economies;
- The *Tariff Profiles* provide information on the market access situation of members, observers and other selected economies;
- The *Services Profiles* provide detailed statistics on key infrastructure services (transportation, telecommunications, finance and insurance) for selected economies;
- The *Time Series* section allows an interactive data retrieval of international trade statistics.

[Trade Profiles](#)

[Tariff Profiles](#)

[Services Profiles](#)

[Time Series on international trade](#)

[Statistics gateway](#)

[Contact Statistics](#)

[Contact Tariffs](#)

[Copyright:](#)

Permission to make digital or hard copies of any information contained in these Web pages is granted for personal or classroom use, without fee and without formal request.

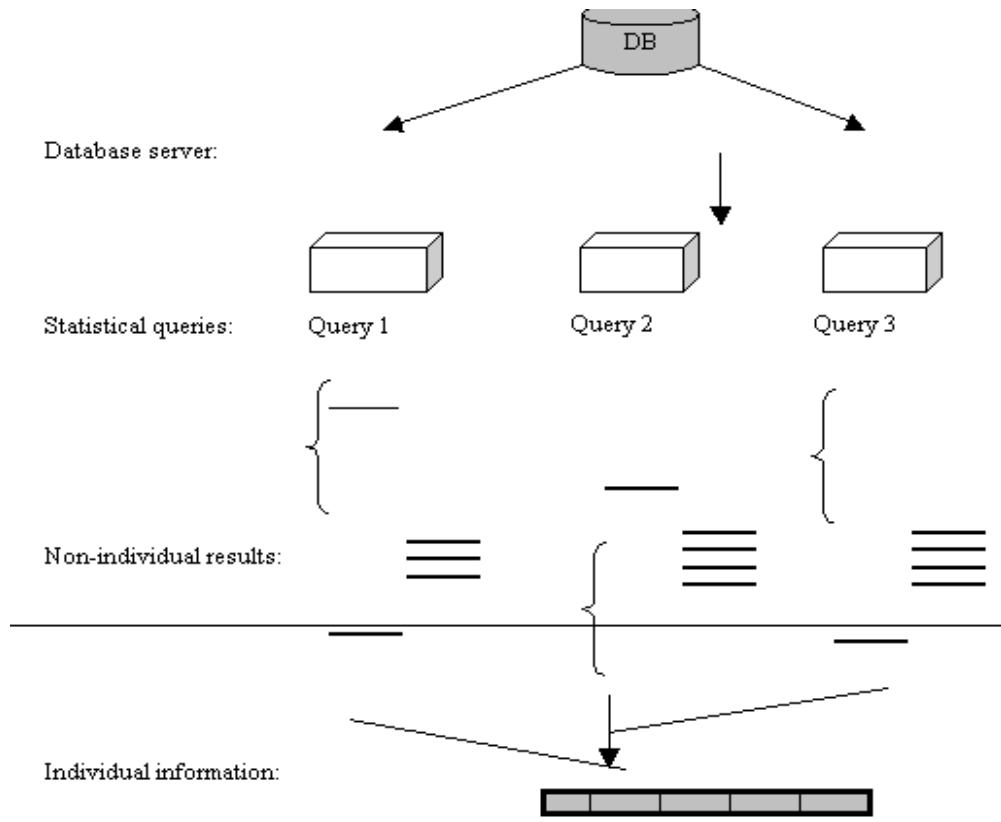
[Full citation and copyright](#)

Hình 5.1. Website cơ sở dữ liệu thống kê của WTO

5.1.3 Vấn đề bảo vệ cơ sở dữ liệu thống kê

Vấn đề chính trong bảo vệ SDB là dàn xếp giữa các yêu cầu cá nhân và quyền của các tổ chức để biết và xử lý thông tin. Nghĩa là bắt buộc phải chia sẻ thông tin thống kê để các tổ chức có thể biết và xử lý thông tin thống kê, trong khi vẫn đảm bảo giữ bí mật các thông tin của cá nhân trong SDB.

Nói đến bảo vệ một SDB là nói đến việc ngăn chặn hay tránh *khả năng suy diễn thống kê* (*statistical inference*). *Suy diễn* trong một SDB có nghĩa là có thể thu được các thông tin bí mật trong các thực thể đơn lẻ, bằng cách lợi dụng các câu truy vấn thống kê.



Hình 5.2. Tần công suy diễn thống kê

Biện pháp an toàn đầu tiên là xây dựng một *bộ lọc thống kê* (*statistical filter*), chỉ cho phép các câu truy vấn thống kê và ngăn chặn truy nhập trực tiếp vào các thực thể xác định trong SDB. Tuy nhiên, bộ lọc thống kê không đủ ngăn chặn suy diễn. Các thông kê đưa ra thường chứa một dấu vết về dữ liệu và sử dụng nó khi tính toán, người sử dụng có kỹ năng có thể thu được các thông tin không được phép. Ví dụ, trước tiên người sử dụng đưa ra câu truy vấn về mức lương trung bình của các nữ nhân công trong một bộ phận nào đó, sau đó đưa ra câu truy vấn tiếp theo về số lượng các nữ nhân công. Nếu phép tính này trả lại giá trị 1 thì người sử dụng thu được (suy diễn) lương của nữ nhân công này thông qua các câu truy vấn thống kê hợp lệ (đây chính là các câu truy vấn mà bộ lọc cho phép đi qua).

Việc phát triển một kỹ thuật kiểm soát suy diễn hiệu quả phụ thuộc vào các kiến thức mà nhà phát triển có được, kiến thức này bao hàm các kỹ thuật

mà một *snooper* (kẻ tấn công theo kiểu đánh hơi) đã sử dụng để tấn công vào SDB. Để đối phó với các kỹ thuật tấn công tinh vi, các cơ chế bảo vệ cũng cần tinh xảo hơn. Hơn nữa, cần kết hợp các kỹ thuật nhằm ngăn chặn suy diễn trên SDB, do đó cần thiết kế các kỹ thuật đơn lẻ để đối phó lại các kiểu tấn công cụ thể.

5.2 CÁC ĐẶC TRƯNG CỦA CƠ SỞ DỮ LIỆU THỐNG KÊ

Trong bảo vệ chống suy diễn, việc thiết kế và thực thi các kỹ thuật cho mục đích này là một nhiệm vụ phức tạp và nảy sinh nhiều vấn đề cần quan tâm. Sau đây, chúng ta xét một số đặc trưng cần bảo vệ của cơ sở dữ liệu thống kê:

- Các SDB có thể *trực tuyến* (*online*): trong đó người sử dụng nhận được các phản hồi thời gian thực cho các câu truy vấn thống kê của mình.
- Các SDB có thể *ngoại tuyến* (*offline*): trong đó người sử dụng không biết khi nào các thống kê của họ được xử lý, việc SDB bị lộ sẽ khó khăn. Tương tự, các SDB có thể *động* hoặc *tĩnh*.
- Các *SDB tĩnh*: là SDB không thay đổi trong suốt thời gian tồn tại của chúng (ví dụ, không xảy ra các thao tác chèn hoặc xoá trong cơ sở dữ liệu điều tra dân số) và các thay đổi chỉ được đưa vào trong các cơ sở dữ liệu tĩnh mới tạo ra.
- Các *SDB động*: thay đổi liên tục theo sự thay đổi của dữ liệu thực, cho phép sửa đổi, nghĩa là được phép chèn hoặc xoá các thực thể để phản ánh các thay đổi động của thế giới thực (ví dụ các cơ sở dữ liệu nghiên cứu trực tuyến, lớp học trực tuyến khi bổ sung thành viên,...). Việc bảo vệ một SDB động phức tạp hơn nhiều, vì cơ sở dữ liệu thường xuyên được bổ sung thêm các thông tin, do đó cần thiết kế các kỹ thuật đặc biệt. Ví dụ, một người sử dụng yêu cầu tính tổng số lượng của các cá nhân trong SDB (đây là các cá nhân có các đặc điểm cụ thể nào đó) trước và sau khi chèn thêm một cá nhân (I) vào SDB, đồng thời cá nhân (I) cũng có các đặc điểm như trên. Do

vậy, người dùng này có thể suy diễn lương của I bằng cách lấy giá trị tổng thứ hai trừ đi giá trị tổng ban đầu.

- *SDB tập trung*: Các hệ CSDL tập trung chạy trên máy đơn và không trao đổi với các máy khác.
- *SDB phân tán*: Dữ liệu phân tán trên các site hoặc trên các phần trong một cơ quan cho phép các dữ liệu thường trú tại nơi chúng được sinh ra nhưng vẫn có thể truy xuất chúng từ các site khác hay các phần khác. Việc lưu nhiều bản sao của CSDL trên các site khác nhau cho phép các tổ chức lớn vẫn có thể tiếp tục hoạt động khi một hay một vài site bị sự cố. Hệ CSDL phân tán được phát triển để quản lý dữ liệu phân tán, trên phương diện địa lý hay quản trị, trải rộng trên nhiều hệ CSDL.

Trong trường hợp này, các kiểm soát suy diễn phức tạp hơn nhiều, do phải áp dụng các kiểm soát tại từng địa điểm và phải kết hợp quản lý các hồ sơ của người dùng. Các SDB có thể hướng các ứng dụng đơn lẻ, hoặc có thể phục vụ như là một tập các ứng dụng hỗn tạp. Việc bảo vệ trong các ứng dụng hỗn tạp rõ ràng là rất phức tạp, do tồn tại một số lượng lớn các kiểu ứng dụng khác nhau tương tác với SDB.

5.3 CÁC KHÁI NIỆM CƠ BẢN

5.3.1. Lộ chính xác và lộ từng phần

Ta nói rằng, một SDB bị làm lộ khi một người dùng thông qua một hay nhiều câu truy vấn và thông qua thu thập thông tin, anh ta có thể suy diễn ra giá trị của các trường cụ thể trong cơ sở dữ liệu. Sự làm lộ này được chia thành hai loại: *lộ chính xác* (*exact compromise*) và *lộ từng phần* (*partial compromise*).

Giả sử, A_i là một thuộc tính không số bí mật (*non-numerical confidential attribute*), hoặc là một thuộc tính số (*numerical attribute*) và x_j là bản ghi mô tả một thực thể trong SDB. Tuỳ thuộc vào kiểu thông tin (mà người sử dụng

thu được) trên A_i để phân biệt *lộ chính xác* và *lộ từng phần*, chúng được định nghĩa như sau:

- *Lộ chính xác (exact compromise)*: Lộ chính xác xảy ra khi người sử dụng thông qua một hoặc nhiều truy vấn thống kê có thể suy diễn ra: thuộc tính A_i có giá trị 1 nếu A_i là một thuộc tính không số-thuộc tính logic, hoặc giá trị chính xác của A_i nếu A_i là một thuộc tính số, với bản ghi x_j trong SDB.
- *Lộ từng phần (partial compromise)*: Lộ từng phần xảy ra khi người sử dụng thông qua một hoặc nhiều truy vấn thống kê có thể suy diễn ra: thuộc tính A_i có giá trị 0 nếu A_i là một thuộc tính không số, hoặc thu được một ước lượng \hat{A} của giá trị thực của thuộc tính A_i nếu A_i là một thuộc tính số, đồng thời tương quan của ước lượng này thoả mãn quan hệ:

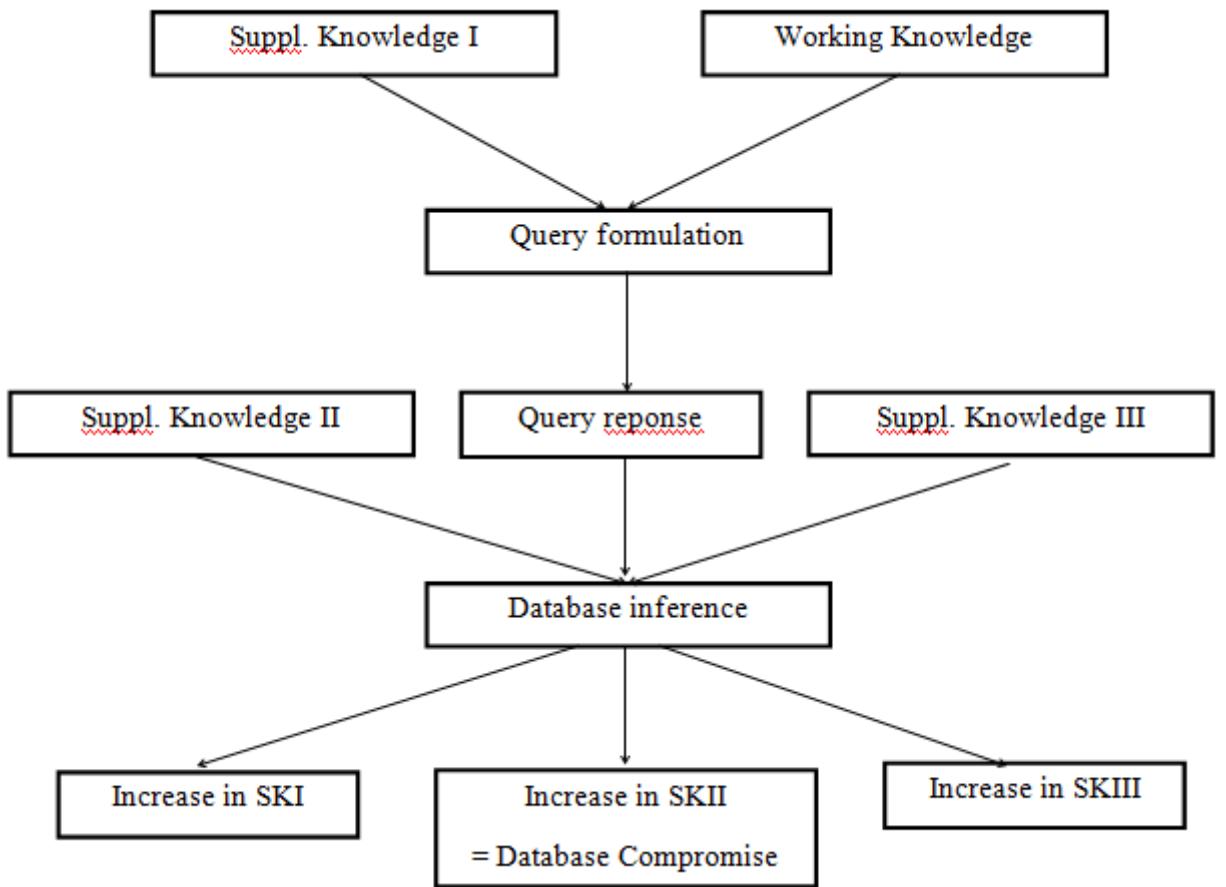
$$Var(\hat{A}) = \left| \hat{A} / A \right| < k^2$$

với bản ghi x_j trong SDB, và k là một tham số do người quản trị cơ sở dữ liệu (SDBA) định nghĩa.

5.3.2 Kiến thức làm việc và kiến thức bổ sung

- *Kiến thức làm việc (Working Knowledge)*: là tập các mục thông tin liên quan đến các giá trị thuộc tính trong SDB và các kiểu thống kê có sẵn trong SDB.
 - *Kiến thức bổ sung của người sử dụng (Supplementary Knowledge)*: là tập các mục thông tin (thường không có trong SDB) mà một người sử dụng có thể có được. Người sử dụng có thể có kiến thức bổ sung về các cá nhân được biểu diễn trong SDB. Họ hoàn toàn có thể lợi dụng kiến thức này cho các mục đích suy diễn. Việc suy diễn sẽ dễ dàng hơn khi người sử dụng có được các thông tin bên lề, giúp họ làm sáng tỏ các kết quả thống kê. Ví dụ, giả thiết rằng chỉ có một nam nhân công làm việc trong một bộ phận cụ thể nào đó. Một câu truy vấn thống kê về tổng số lương của các nam nhân công trong bộ phận này có thể

suy ra lương của nhân công nếu người sử dụng biết được tên của nhân công này.



Hình 5.3. Làm lộ SDB và kiến thức bổ sung

5.3.3 Công thức đặc trưng

Công thức đặc trưng được ký hiệu bởi một chữ cái viết hoa (A, B, C, \dots), đây là một công thức lôgíc, trong đó các giá trị thuộc tính được kết hợp với nhau thông qua các toán tử Boolean như OR, AND, NOT (\vee, \wedge, \neg) được liệt kê theo thứ tự ưu tiên tăng dần. Một ví dụ về công thức đặc trưng như sau:

$$A = (Sex=F) \wedge ((Dept-Code=Dept1) \vee (Dept-Code=Dept2)) \wedge (Birth-Year < 1965)$$

Câu truy vấn này xác định tất cả các nữ nhân công của các phòng 'Dept1' và 'Dept2' sinh trước năm 1965.

5.3.4 Tập truy vấn

Tập truy vấn của một công thức đặc trưng C là tập các bản ghi trong SDB thỏa mãn C, ký hiệu: X(C).

Ví dụ, với SDB về công nhân sau:

ID	Ten	ChucVu	Phong	Tuoi	GioiTinh	Luong
01	Phương	Nhân viên	Maketing	30	F	3700
02	Tuyền	Trưởng phòng	Kế hoạch	42	M	6500
03	Phương	Nhân viên	Kế hoạch	19	M	3800
04	Huyền	Giám sát viên	Maketing	28	F	4500
05	Phương	Nhân viên	Maketing	25	M	2500
06	Thắng	Phó phòng	Tài vụ	39	M	3900

Bảng 5.3. Ví dụ SDB công nhân

Công thức đặc trưng là:

$$C = (Phong = "Maketing") \wedge (Tuoi > 25)$$

Khi đó, tập truy vấn của công thức đặc trưng C là X(C) được biểu diễn dưới đây:

ID	Ten	ChucVu	Phong	Tuoi	GioiTinh	Luong
01	Phương	Nhân viên	Maketing	30	F	3700
04	Huyền	Giám sát viên	Maketing	28	F	4500

Bảng 5.4. Ví dụ về X(C) của SDB công nhân

All là một đặc trưng đặc biệt, tập truy vấn của nó chứa tất cả các bản ghi của SDB. Nói chung, quan hệ $X(\text{All}) \supseteq X(C)$ đúng với mọi công thức đặc trưng C , có nghĩa là tập truy vấn của một đặc trưng C bất kỳ là một tập con của toàn bộ các bản ghi trong SDB.

5.3.4 Các truy vấn thống kê

Các câu truy vấn thống kê chủ yếu mà người dùng sử dụng hiệu quả khi yêu cầu thông tin trong một SDB là: *Count*, *Sum*, *Rfreq*, *Avg*, *Median*, *Max* và *Min*. Chúng được minh họa chi tiết như sau:

- $\text{Count}(C) = |X(C)|$

Được sử dụng để đếm các bản ghi có trong tập truy vấn thỏa mãn công thức đặc trưng C cho trước.

- $\text{Sum}(C, A_j) = \sum_{i \in X(C)} x_{ij}$

Được sử dụng để tính tổng các giá trị của một thuộc tính số A_j cho trước đối với tất cả các bản ghi của tập truy vấn $X(C)$.

- $\text{Rfreq}(C) = |X(C)|/N$, N là tổng số bản ghi của SDB

Được sử dụng để tính tần số tương đối của $X(C)$ với tổng số bản ghi trong SDB.

- $\text{Avg}(C, A_j) = \text{Sum}(C, A_j)/|X(C)|$

Được sử dụng để tính giá trị trung bình của một thuộc tính số A_j trong tập truy vấn $X(C)$.

- $\text{Max}(C, A_j) = \max_{i \in X(C)} x_{ij}$

Được sử dụng để xác định giá trị lớn nhất của một thuộc tính số A_j trong tập truy vấn $X(C)$.

- $\text{Min}(C, A_j) = \min_{i \in X(C)} x_{ij}$

Được sử dụng để xác định giá trị nhỏ nhất của một thuộc tính số A_j trong một tập truy vấn $X(C)$.

- $\text{Median}(C, A_j) = \lceil |X(C)|/2 \rceil$

Được sử dụng để tính giá trị trung bình trong một tập các giá trị số (được sắp xếp theo thứ tự tăng dần) của một thuộc tính A_j . Giá trị trung bình của một thuộc tính là giá trị được xác định tại vị trí $\lceil |X(C)|/2 \rceil$ trong tập truy vấn gắn liền với C .

5.3.5 Khái niệm bậc

Với các câu truy vấn thông kê có thể định nghĩa khái niệm *bậc*. Một thông kê bao gồm các giá trị của m thuộc tính khác nhau)được gọi là thông kê bậc m . Ví dụ, thông kê $\text{Count}((\text{Sex} = F) \wedge (\text{Dept-Code} = \text{Dept1}))$ là một thông kê bậc 2, với 2 thuộc tính Sex và Dept-Code . $\text{Count}(\text{All})$ chỉ là một thông kê bậc 0.

5.3.6 Thông kê nhạy cảm

Một khái niệm quan trọng trong các SDB là khái niệm thông kê nhạy cảm. *Thông kê nhạy cảm* là một thông kê có thể được sử dụng để nhận dạng thông tin bí mật về một cá nhân được biểu diễn trong SDB.

Thông kê được tính toán trên một *thuộc tính bí mật* trong tập truy vấn là thông kê nhạy cảm. Ví dụ, khi ta dùng truy vấn $\text{COUNT}(\text{AGE} > 50)$ để đếm số công nhân có tuổi lớn hơn 50 chẳng hạn, và số lượng này chỉ là một, tiếp theo ta có thể dễ dàng tìm ra lương của người công nhân này bằng câu truy vấn $\text{SUM}(\text{Salary}, \text{age} > 50)$, như vậy truy vấn SUM trên là *thông kê nhạy cảm*, và *thông tin bí mật* ở đây chính là *luong (salary)*. Do đó, nên hạn chế các thông kê nhạy cảm, có nghĩa là không nên để lộ chúng.

5.4 CÁC TẤN CÔNG VÀO CƠ SỞ DỮ LIỆU THỐNG KÊ

Như đã đề cập, vấn đề lớn nhất với các SDB là tấn công suy diễn thông kê. Một số tấn công suy diễn vào SDB bao gồm: tấn công trực tiếp, tấn công dựa vào đếm, tấn công trình theo dõi, tấn công hệ tuyến tính.

Trong *tấn công trực tiếp*, người sử dụng có thể ngẫu nhiên hay cố ý thực hiện các câu truy vấn trực tiếp vào các trường dữ liệu nhạy cảm trong SDB.

Chẳng hạn câu truy vấn vào *tên* liên quan đến mục *luong* trong SDB về công nhân sau:

Select Ten from NhanVien where Luong > 5000;

Giải pháp để chống kiểu tấn công này là sử dụng các bộ lọc thống kê nhằm loại đi các truy vấn liên quan đến từng mục dữ liệu riêng trong SDB.

Với dạng *tấn công dựa vào đếm*, kẻ tấn công trước tiên thực hiện thống kê *count* để đếm số lượng các bản ghi trong SDB thỏa mãn một công thức đặc trưng nào đó, tiếp theo thực hiện các thống kê như *Sum, Min, Max, Average*. Loại tấn công này dễ dàng thu được thông tin bí mật trong trường hợp kết quả của thống kê *count* là rất nhỏ, chẳng hạn chỉ có một (hoặc hai) bản ghi.

Ví dụ: Kẻ tấn công thực hiện liên tiếp hai thống kê sau

Count((ChucVu = "Truong phong") \wedge (Phong = "Ke hoach")) = 1

Sum(Luong, C)

Từ kết quả của hai thống kê này, kẻ tấn công có thể thu được lương của người trưởng phòng của phòng Kế hoạch.

Tấn công trinh theo dõi và tấn công hệ tuyến tính là hai loại tấn công mạnh vào SDB, phần Kỹ thuật kiểm soát dựa vào hạn chế sẽ được trình bày sau, để thấy rõ hơn sức mạnh của hai loại tấn công này.

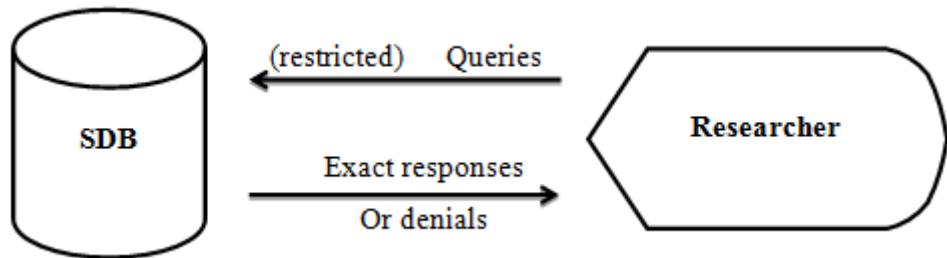
5.5 CÁC KỸ THUẬT CHỐNG SUY DIỄN

Mục đích của các kỹ thuật chống suy diễn (còn được gọi là các kiểm soát suy diễn) là ngăn chặn người sử dụng suy diễn các thông tin bí mật liên quan đến các cá nhân được biểu diễn trong SDB.

Từ sự phân loại tổng quát các kỹ thuật chống suy diễn do Denning và Schlorer (1983) và Adam, Wortmann (1989) đưa ra, ta có thể phân loại các kỹ thuật này thành: *kỹ thuật khái niệm*, *kỹ thuật dựa vào hạn chế* (*restriction-based technique*) và *kỹ thuật dựa vào gây nhiễu* (*perturbation-based technique*).

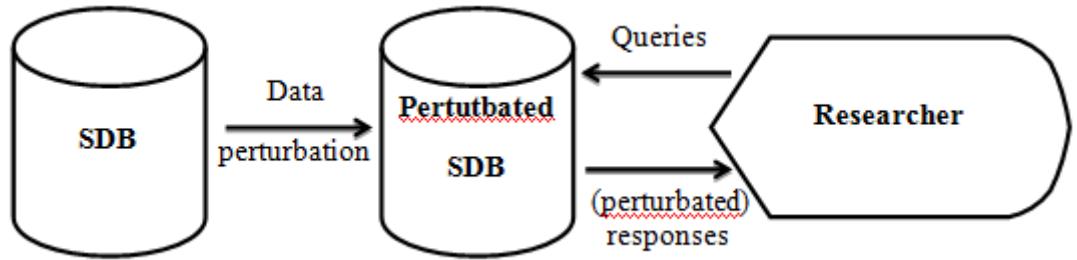
- *Các kỹ thuật khái niệm*: giải quyết vấn đề suy diễn tại mức khái niệm, xử lý ở mô hình dữ liệu khái niệm của SDB.

- *Các kỹ thuật dựa vào hạn chế*: chống suy diễn bằng cách hạn chế một số câu truy vấn thông kê (chẳng hạn như các câu truy vấn mà tập truy vấn của nó chứa một số lượng nhỏ/lớn các bản ghi, hoặc các tập truy vấn chứa một vài bản ghi chung). Mặc dù chống suy diễn được một phần, nhưng nhược điểm của kỹ thuật này là hạn chế số lượng lớn các câu truy vấn thông kê và do đó làm hạn chế phần lớn khả năng sử dụng của SDB.

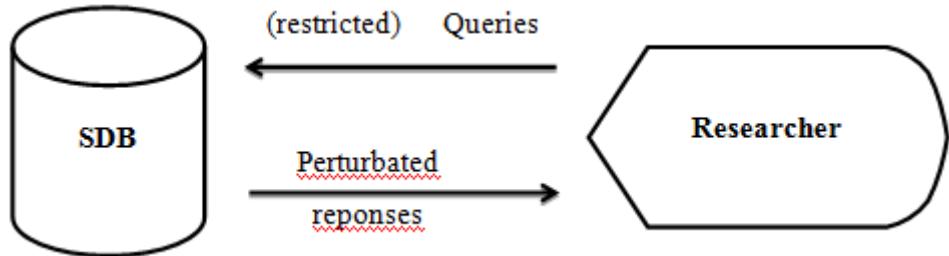


Hình 5.4. Kỹ thuật dựa vào hạn chế

- *Các kỹ thuật dựa vào gây nhiễu*: chống suy diễn bằng cách đưa ra các sửa đổi đối với thông tin được dùng khi trả lời các câu truy vấn thông kê. Việc sửa đổi có thể được thực hiện với dữ liệu lưu giữ trong SDB, hoặc không sửa đổi dữ liệu trong SDB nhưng sửa đổi kết quả tính toán trước khi chuyển tới người sử dụng. Ưu điểm của kỹ thuật này là chống được suy diễn nhưng hạn chế là đưa ra các kết quả truy vấn không chính xác. Vấn đề chủ yếu đối với các kỹ thuật này là *độ lệch (bias)*. Do việc sửa đổi gây mất mát thông tin, chính vì vậy phải sửa đổi sao cho vẫn đảm bảo được *tính tương thích* của các kết quả.



Hình 5.5. Kỹ thuật gây nhiễu dữ liệu



Hình 5.6. Kỹ thuật gây nhiễu đầu ra

Bây giờ chúng ta sẽ đi vào tìm hiểu từng kỹ thuật kiểm soát suy diễn cụ thể:

5.5.1 Các kỹ thuật khái niệm

Trong phần này chúng ta sẽ mô tả các kỹ thuật khái niệm cơ bản để chống suy diễn, cụ thể là: *Mô hình lưới* (do Denning và Schlorer đề xuất, 1983) và kỹ thuật *Phân hoạch khái niệm* (do Chin và Ozsoyoglu đề xuất, 1981) để phân hoạch các thực thể riêng lẻ trong SDB thành các phần (population) trong giai đoạn thiết kế khái niệm của SDB.

5.5.1.1 Mô hình lưới

Mô hình khái niệm cung cấp một nền tảng để khám phá ra các vấn đề an toàn ở mức mô hình dữ liệu khái niệm. Một cách tiếp cận phổ biến cho mô hình khái niệm là *mô hình lưới*. Mô hình lưới thể hiện một nền tảng tốt để tìm hiểu và khám phá ra các vấn đề an toàn trong các SDB, nhưng lại đưa ra quá nhiều ràng buộc cho người sử dụng. Mô hình lưới mô tả các thông tin SDB dưới dạng bảng ở các mức gộp khác nhau. Mô hình này được quan tâm, bởi

vì xuất phát từ thực tế thông tin thống kê được cung cấp ở các mức gộp khác nhau, do đó có thể đưa ra thông tin dư thừa và người dùng có thể sử dụng thông tin dư thừa này để suy diễn ra các thông tin nhạy cảm. Nếu thông tin bí mật bị cấm ở mức chi tiết, thì thông tin đó có thể bị khám phá bằng nhiều thông tin gộp.

Đây là một mô hình ở mức khái niệm, nó cung cấp biểu diễn ở mức khái niệm đối với các bản ghi được lưu giữ trong SDB, dựa vào *cấu trúc lưới* với các bảng m chiều (m thuộc tính) được tổ chức tại các mức trùu tượng khác nhau.

NGP Table				
NamSinh	GioiTinh	Phong		
		Phong1	Phong2	Phong3
1941-1951	M	10	12	0
	F	1	0	3
1952-1962	M	12	10	5
	F	20	2	8
>1962	M	15	0	1
	F	20	10	0

Hình 5.7. Thống kê vĩ mô về công nhân theo năm sinh, giới tính và phong

Một ví dụ về bảng m -chiều được minh họa trong hình 5.7, đó là một bảng 3 chiều chỉ ra thống kê *đếm* (count) số lượng nhân công với 3 thuộc tính “NamSinh”, “GioiTinh” và “Phong”. (Dạng bảng thường được dùng với các SDB có mục đích đặc biệt như thống kê điều tra dân số của Mỹ, gồm một bảng 2 chiều với các thống kê Count và Sum).

Nói chung, bảng m chiều biểu diễn một tập các thống kê tương quan *bậc* m , được tính toán trên m ($m < M$) thuộc tính của SDB, A_1, \dots, A_M . Mỗi thuộc tính A_i có $|A_i|$ giá trị, vậy tổng số các phần tử của bảng S_m được tính như sau :

$$S_m = \prod_{i=1}^m |A_i|$$

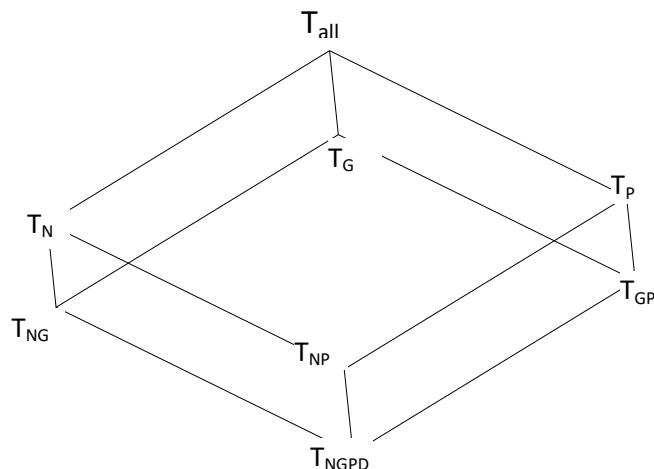
Các thống kê có nguồn gốc từ bảng là $2^{sm}-1$.

- *Dạng bảng (tabular form):*

Dạng bảng không tương ứng với tổ chức dữ liệu thực trong SDB mà nó chỉ đưa ra các thống kê. Đúng hơn, nó là một khung nhìn của SDB được tính toán bằng cách *gộp* (*microaggregation*) dữ liệu thực từ mô hình quan hệ. Xét M thuộc tính của SDB, A_1, \dots, A_M , mỗi thuộc tính có $\{1 \dots |A_i|\}$ giá trị (gọi là bộ giá trị thuộc tính). Mỗi bộ giá trị thuộc tính định rõ các thực thể được nhóm thành các lớp hay loại nào, để dễ dàng tính được các thống kê trên. Dữ liệu về các cá thể của SDB sẽ được mô tả trong bảng *M-chieu* trong đó, các thực thể cùng loại sẽ được phân vào một ô trong bảng này. Mỗi ô trong bảng *M-chieu* sẽ tương ứng với một tập truy vấn được xác định qua M thuộc tính phân biệt. Một SDB với M thuộc tính sẽ có 2^M bảng *m-chieu*, như ở ví dụ trên $M=3$ do đó ta có $2^3=8$ bảng *m-chieu* ($m=0,1,2,3$).

- *Cấu trúc lưới:*

Tập các bảng *m-chieu* đó (liên quan đến một thống kê đã đưa ra, như ở ví dụ trên ta có thống kê *count*) tạo thành một cấu trúc lưới. Lưới được xây dựng thông qua *cơ chế gộp* đối với một thuộc tính, thu được các bảng có kích cỡ nhỏ hơn M , cho đến khi thu được một bảng 0 chiều biểu diễn thống kê *Count* (điều này thể hiện rằng thống kê *count* đã được tính toán trên toàn bộ SDB).



Hình 5.8 Lưới bảng trên các thuộc tính *NamSinh*, *GioiTinh*, *MaPhong*

Ví dụ, bằng cách gộp bảng $M=3$ chiều trong hình 5.6 với các thuộc tính *Birth-Year*, *Sex* và *Dept-Code*, chúng ta thu được các bảng 2 chiều *NG*, *NP* và *GP* (được mô tả trong hình 5.7) cho thống kê *Count*.

NG Table			
NamSinh	GioiTinh		
	M	F	
1941-1951	22	4	
1952-1962	27	30	
>1962	16	30	

NP Table			
NamSinh	Phong		
	Phong1	Phong2	Phong3
1941-1951	11	12	3
1952-1962	32	12	13
>1962	35	10	1

GP Table			
GioiTinh	Phong		
	Phong1	Phong2	Phong3
M	37	22	6
F	41	12	11

Hình 5.9. Các bảng 2-chiều

Các bảng 1-chiều thu được là :

Giới tính	
M	F
65	64

Năm sinh	
1941-1951	26
1952-1962	58
>1962	46

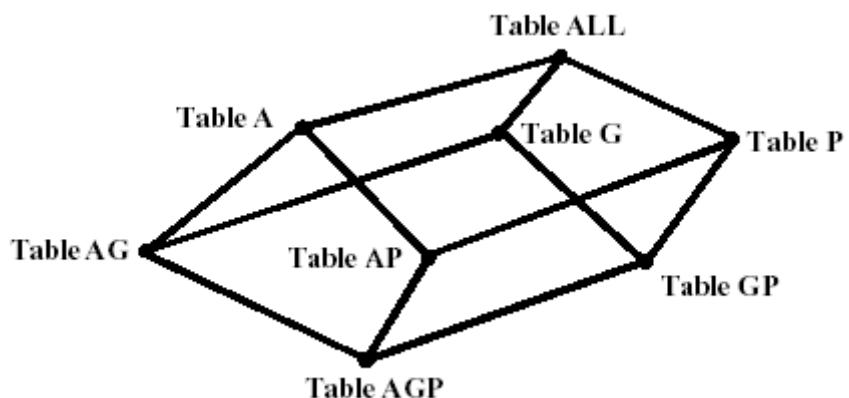
Mã phòng		
Phòng1	Phòng2	Phòng3
78	34	17

Hình 5.10. Các bảng 1-chiều

Bảng 0-chiều thu được là : 129

- Một ví dụ khác về mô hình lưới:

Ngoài ví dụ trên về cơ sở dữ liệu thống kê, để hiểu hơn về mô hình lưới ta đưa thêm một ví dụ sau:



Hình 5.11. Mô hình lưới 2

Xét một cơ sở dữ liệu của trường đại học với các thuộc tính *tuổi -Age*, *giới tính - Gender*, và *vị trí - Position* (ký hiệu 3 thuộc tính là A, G , P). Mô hình lưới tương ứng được chỉ ra ở hình trên. Một cách chi tiết nhất để thể hiện cơ sở dữ liệu này là biểu diễn dưới *dạng bảng* bao gồm bảng 3-chiều AGP với các chiều A, G và P. Trong bảng 3-chiều, chứa một ô phần tử cá biệt (nhạy cảm) là ô với A =42, G = M và P = Professor (count =1). Dạng bảng này có thể được gộp thành 3 bảng 2- chiều.

1. Bảng AG, được tạo từ bảng AGP khi gộp theo chiều P.
2. Bảng AP, được tạo từ bảng AGP khi gộp theo chiều G
3. Bảng GP, được tạo từ bảng AGP khi gộp theo chiều A

Việc gộp có thể được thực hiện đến khi chỉ còn một bảng 0-chiều (Table All). Bảng này chỉ chứa một ô, cung cấp thông kê cho toàn bộ cơ sở dữ liệu. Lưu ý rằng tập tất cả các bảng 2-chiều đôi khi có thể khám phá ra thông kê ô phần tử nhạy cảm của bảng 3 chiều.

Table AGP

Position	Gender	Age			
		0-20	21-45	46-65	>65
Professor	M	0	1	9	0
	F	0	16	0	0
Vice Professor	M	1	20	48	0
	F	1	0	52	0
Others	M	24	2	9	49
	F	26	0	1	51

Table AG:

Gender	Age			
	0-20	21-45	46-65	>65
M	25	23	66	49
F	27	16	53	51

Table AP

Position	Age			
	0-20	21-45	46-65	>65
Professor	0	17	9	0
Vice Professor	2	20	100	0
Others	50	2	10	100

Table GP:

Position	Gender	
	M	F
Professor	10	16
Vice Professor	69	53
Others	84	78

Table A:

Age			
0-20	21-45	46-65	>65
52	39	119	100

Table G

Gender	
M	F
163	147

Table P

Position		
Professor	Vice Professor	Others
26	122	162

Table ALL: 310

Hình 5.12 Các bảng ở các mức gộp khác nhau

Mục tiêu của mô hình lưới là để kiểm soát suy diễn với các thông kê nhạy cảm. Ở đây, chúng ta sẽ giả sử rằng một thông kê nhạy cảm tương ứng với

kích cỡ tập truy vấn bằng 1. Vì vậy ô tương ứng với A =42, G = M và P = Professor là một thông kê nhạy cảm.

Nhận xét:

Mô hình lưới đã được chứng minh là một mô hình an toàn hiệu quả cho nghiên cứu các vấn đề suy diễn và các phương pháp kiểm soát suy diễn. Vì từ mô hình này với nhiều bảng ở các mức gộp khác nhau, ta có thể phân tích các kiểu tấn công suy diễn bằng câu truy vấn COUNT, SUM, AVERAGE,...và các tấn công kiểu kết hợp các câu truy vấn khác nhau để suy diễn ra dữ liệu nhạy cảm...Nhưng mô hình lưới không thể cung cấp tính đầy đủ của cơ sở dữ liệu và không phù hợp với cơ sở dữ liệu động, vì khi cập nhật SDB ta phải cập nhật tất cả các bảng trong mô hình lưới, do đó rất tốn công.

Việc sử dụng mô hình lưới đã cung cấp một khung làm việc hợp lệ cho nghiên cứu và so sánh các kiểm soát suy diễn khác nhau, chẳng hạn như các kiểm soát suy diễn dựa vào việc hạn chế câu truy vấn và sửa đổi dữ liệu. Thông qua việc sử dụng *dạng bảng* của mô hình lưới, chúng ta nghiên cứu các kỹ thuật kiểm soát suy diễn nhằm hạn chế thông kê nhạy cảm

Một thông kê đưa ra tập truy vấn có kích cỡ bằng 1 được gọi là một thông kê nhạy cảm. Khi đó, trong các bảng *m-chiều* với thông kê *count* thì các ô có giá trị 1 được gọi là nhạy cảm. Tương tự như vậy, thông kê *sum* được tính trong các ô bằng 1 đó cũng trở thành nhạy cảm. Bằng cách khai thác mối quan hệ giữa các bảng *m-chiều* trong cấu trúc lưới ở các mức trừu tượng khác nhau, một người dùng có thể khám phá ra các ô nhạy cảm.

5.5.1.2 Phân hoạch khái niệm

Kỹ thuật này do Chin và Ozsoyoglu đề xuất (1981), giải quyết các vấn đề chống suy diễn trong giai đoạn thiết kế khái niệm của SDB, dựa vào việc định nghĩa tập các cá thể của SDB tại mức khái niệm, được gọi là các *lực lượng* (*populations*) và dựa vào các điều kiện cần kiểm tra nhằm tránh suy diễn. Chính xác hơn, *mô hình trừu tượng hóa dữ liệu* (*Data Abstraction - D-A* do Smith đưa ra năm 1977) dựa vào các trừu tượng *gộp* và các trừu tượng *tổng*

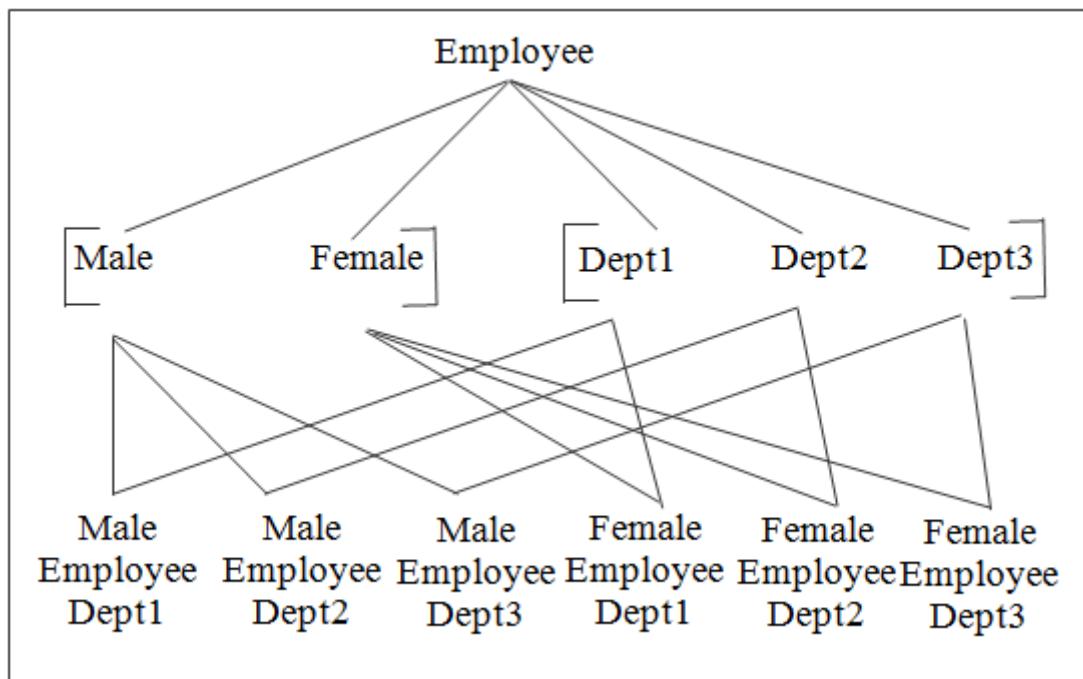
quát hóa để mô hình hóa tập các cá thể của SDB. *Gộp (aggregation)* cho phép biểu diễn *các mối quan hệ* của đối tượng bằng các đối tượng gộp. *Tổng quát hóa (generalization)* cho phép biểu diễn *các lớp* đối tượng thành các đối tượng chung (generic object). Sau đó, thế giới thực được biểu diễn thông qua các *hệ thống phân cấp tổng quát và gộp*, các phần chung của hệ thống này là các đối tượng trừu tượng (được gộp và tổng quát).

Để đáp ứng các mục đích thống kê, các *hệ thống phân cấp tổng quát* được quan tâm nhiều hơn, do chúng cho phép các nhà thiết kế mô tả các cá nhân trong thế giới thực (được biểu diễn trong SDB) tại các mức tổng quát khác nhau. Trong một hệ thống phân cấp tổng quát, có thể định nghĩa các mức đối tượng chung khác nhau, việc phân hoạch các cá nhân tùy thuộc vào các giá trị thuộc tính của chúng. Một tập các cá thể của SDB nắm giữ các thuộc tính chung sẽ được xem là *một lực lượng (population)*. Bằng cách dịch chuyển từ gốc đến các lá trong một hệ thống phân cấp tổng quát, các lực lượng được phân tách thành các lực lượng con và dần dần phân tách thành các lực lượng nguyên tử (*A-population*), đây chính là các lực lượng không thể phân tách được nữa và chúng chính là các lá của hệ thống phân cấp.

Hình 5.13 minh họa mô hình khái niệm của một cơ sở dữ liệu thống kê về công nhân - Employee SDB, trong đó lực lượng Employee được phân tách thành 5 lực lượng con, tùy thuộc vào các thuộc tính "Sex" và "Dept-Code".

Các lực lượng trung gian hình thành hai cụm, *Sex = (Male, Female)* và *Dept-Code = (Dept1, Dept2, Dept3)*. Các lá của cây biểu diễn các lực lượng nguyên tử, các lực lượng nguyên tử này bao gồm các nhân công cùng giới trong từng bộ phận.

Mô hình khái niệm này của một cơ sở dữ liệu thống kê được sử dụng để xác định những thống kê nào có thể được đưa ra và tránh suy diễn. Nói riêng, với mỗi lực lượng của mô hình, nhà thiết kế sẽ xác định các câu truy vấn thống kê nào (như tính tổng - *Sum*, tính trung bình - *Average*) được phép trên các thuộc tính nào của lực lượng. Hơn nữa, từ mô hình này nhà thiết kế có thể xác định xem có áp dụng được *thống kê Count* cho toàn bộ lực lượng được hay không.



Hình 5.13. Mô hình khái niệm áp dụng cho Employee SDB

Để làm tốt hơn việc xác định các yêu cầu an toàn thống kê, người ta đưa ra khái niệm *lực lượng nguyên tử an toàn* (*SA-population*). Một lực lượng nguyên tử an toàn nhóm một hoặc nhiều lực lượng nguyên tử thành một lực lượng lôgíc, mà từ đó không thể đưa ra thông tin thống kê liên quan đến bất kỳ một thuộc tính cụ thể nào. Khái niệm *tập giá trị nguyên tử an toàn* (*security atom value set*) được sử dụng để chỉ ra tập các giá trị thuộc tính cần được bảo vệ, cho các thuộc tính của một lực lượng nguyên tử an toàn.

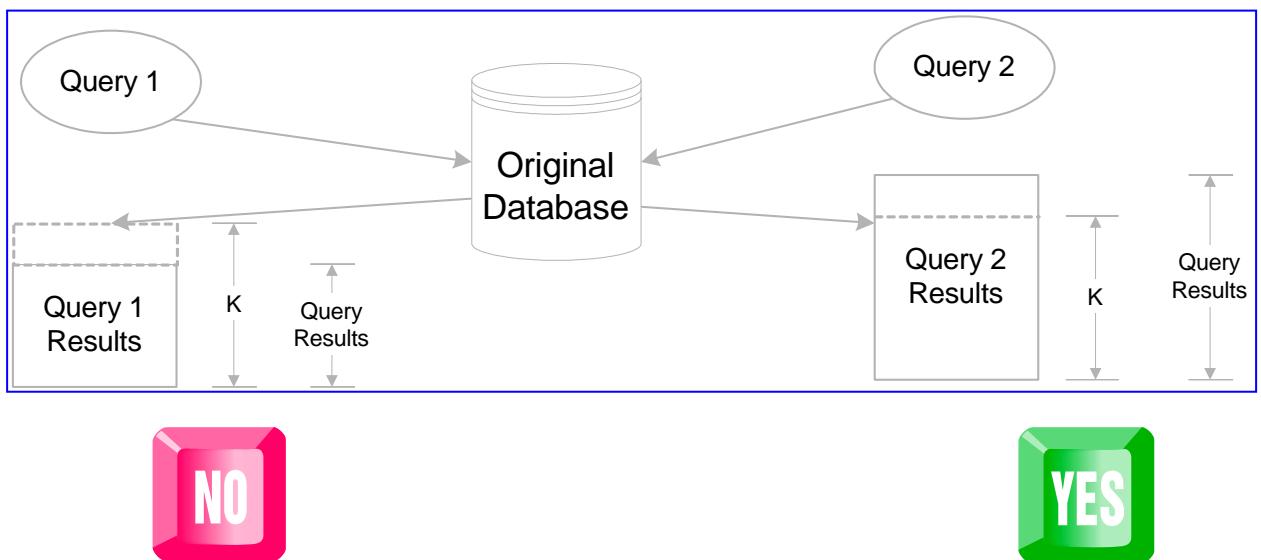
5.5.2 Các kỹ thuật hạn chế

Các kỹ thuật này chống suy diễn bằng cách hạn chế các câu truy vấn thống kê theo một điều kiện hạn chế nào đó, vì thông qua các thông kê này người sử dụng có thể biết được thông tin bí mật về các cá nhân được biểu diễn trong SDB. Nói chung theo cách này, tập truy vấn được đưa ra là một tập con trong các tập truy vấn yêu cầu, có nghĩa là chỉ một số các tập truy vấn tương ứng với các truy vấn thỏa mãn điều kiện hạn chế mới được đưa ra.

Các kỹ thuật dựa vào hạn chế được miêu tả bao gồm:

- Kiểm soát kích cỡ tập truy vấn
- Kiểm soát kích cỡ tập truy vấn mở rộng
- Kiểm soát chồng lấp tập truy vấn
- Kiểm toán
- Phân hoạch
- Giấu ô
- Kỹ thuật kết hợp

5.5.2.1 Kiểm soát kích cỡ tập truy vấn



Hình 5.14. Kỹ thuật kiểm soát kích cỡ tập truy vấn

Trong kỹ thuật kiểm soát kích cỡ tập truy vấn, một thông kê $q(C)$ chỉ được phép nếu tập truy vấn của nó, $X(C)$, thoả mãn quan hệ sau:

$$k \leq |X(C)| \leq N-k$$

$$0 \leq k \leq N/2$$

Với N là số lượng các bản ghi của SDB và $k \geq 0$ là một tham số cố định. Lưu ý rằng, k phải thoả mãn điều kiện $0 \leq k \leq N/2$, để có thể đưa ra các thông kê và không đưa ra thông kê $q(\text{All})$ (thông kê này có thể được tính bằng

$q(\text{All}) = q(C) + q(\neg C)$ cho từng đặc trưng C sao cho $k \leq |X(C)| \leq N-k$. Kiểm soát này ngăn chặn các tấn công đơn giản, dựa vào các tập truy vấn rất nhỏ hoặc rất lớn.

Giả sử, một người dùng biết được một cá nhân cụ thể nào đó thỏa mãn công thức đặc trưng C cho trước. Nếu thông kê $q_1 = \text{Count}(C)$ trả lại giá trị 1, người dùng có thể xác định duy nhất cá nhân này thỏa mãn C trong SDB. Tiếp đó, anh ta có thể khám phá các thông tin phụ liên quan đến cá nhân này, bằng cách đưa ra thông kê $q_2 = \text{Count}(C \wedge C')$, trong đó đặc trưng C' bao gồm các thuộc tính bổ sung quan trọng của cá nhân đó. Nếu q_2 trả lại giá trị 1 tức là cá nhân cũng thỏa mãn C' , ngược lại nếu q_2 trả lại giá trị 0, cá nhân không thỏa mãn C' .

Bằng cách áp dụng kiểm soát kích cỡ tập truy vấn, các thông kê q_1 và q_2 sẽ không được đưa ra, do đó người sử dụng không thể suy diễn thông tin bí mật về các cá nhân. Cũng nên hạn chế các tập truy vấn rất lớn, bởi vì người sử dụng có thể thu được thông tin trên một tập truy vấn nhỏ (được xác định bởi công thức đặc trưng C), bằng cách sử dụng $q(\neg C)$ là một tập truy vấn có kích cỡ $N-k$, trong đó k là kích cỡ của $X(C)$.

Tuy nhiên, kiểm soát này chỉ ngăn chặn được các tấn công đơn giản, khó có thể ngăn chặn được các tấn công phức tạp, chẳng hạn như: tấn công Trình theo dõi, tấn công Hệ tuyến tính.

5.5.2.1.1 Tấn công trình theo dõi (Tracker)

(Schlorer, 1980; Denning, 1982)

Trình theo dõi là một tập các công thức đặc trưng, có thể được sử dụng để đưa thêm bản ghi vào các tập truy vấn kích cỡ nhỏ, làm cho kích cỡ của chúng nằm trong khoảng $[k, N-k]$. Thông qua các trình theo dõi có thể tính toán được các thông kê bị hạn chế.

Kiểu tấn công Trình theo dõi 1:

Giả sử C là công thức đặc trưng thỏa mãn $|X(C)| < k$, khi đó các thông kê trên C đều bị cấm bởi Kiểm soát kích cỡ tập truy vấn. Đồng thời, công thức đặc trưng C có dạng:

$$C = A \wedge B$$

Kẻ tấn công sẽ đưa ra một công thức đặc trưng T như sau:

$$T = (A \wedge \neg B)$$

thỏa mãn: $k \leq |X(T)| \leq N - k$

Khi đó, các thông kê trên T không bị cấm bởi kiểm soát kích cỡ tập truy vấn.

Từ đó, kẻ tấn công có thể tính được các thông kê trên C nhờ T như sau:

$$Q(C) = Q(A \wedge B) = Q(A) - Q(A \wedge \neg B)$$

$$\text{hay } Q(C) = Q(A) - Q(T)$$

Để làm sáng tỏ, chúng ta xem xét một ví dụ rất đơn giản về kiểu tấn công thứ nhất dựa vào Trình theo dõi. Giả thiết rằng, đặc trưng:

$C = (Dept-Code = Dept1 \wedge Sex = F \wedge Birth-Year = 1951)$ xác định duy nhất nhân công Brown trong Employee SDB và cơ chế kiểm soát kích cỡ tập truy vấn có tham số $k = 3$. Do đó, thông kê Count (C) không được phép vì $|Count(C)| = 1$.

Ta có thể thu được một Trình theo dõi bằng cách định nghĩa công thức đặc trưng $A = (Sex = F)$, $B = (Dept-Code = Dept1 \wedge Birth-Year = 1951)$ và $T = (A \wedge \neg B)$. Và T chính là một trình theo dõi.

Một trình theo dõi như vậy được gọi là *trình theo dõi cá nhân*, bởi vì nó được xây dựng bằng cách sử dụng kiến thức của người dùng về một cá nhân cụ thể nào đó được biểu diễn trong SDB, cho phép người dùng thu được thông tin phụ bí mật về cá nhân này (ví dụ, nhân công Brown). Trong trường hợp của chúng ta, thông kê bị ngăn cấm:

$q(C) = Count(C) = (Sex = F \wedge Dept-Code = Dept1 \wedge Birth-Year = 1951)$
có thể được tính toán, bằng cách sử dụng trình theo dõi theo cách sau đây:

$$\begin{aligned}Count(A \wedge B) &= Count(A) - Count(A \wedge \neg B) \\ \Leftrightarrow Count(C) &= Count(A) - Count(T)\end{aligned}$$

Hơn nữa có thể suy diễn thông tin bí mật về lương của Brown, bằng cách đưa ra câu truy vấn sau đây :

$$Count(C \wedge Salary \geq 20) = Count(T \vee A \wedge Salary \geq 20) - Count(T)$$

Nếu kết quả của câu truy vấn này là 1, người sử dụng suy diễn thông tin về lương của Brown.

Kiểu tấn công Trình theo dõi 2:

Giả thiết, cần tính $Count(C)$, với $Count(C) < k$, nên thống kê trên C bị cấm bởi Kiểm soát kích cỡ tập truy vấn.

Kẻ tấn công sẽ chọn một công thức đặc trưng T thỏa mãn: $k \leq |X(T)|$, $|X(\neg T)| \leq N-k$. Sau đó tính $Q(D) = Q(\text{All}) = Q(T) + Q(\neg T)$ (vì $Q(\text{All})$ bị cấm, nên được tính thông qua $Q(T)$ và $Q(\neg T)$). Khi đó có thể tính gián tiếp các thống kê trên C thông qua T như sau :

$$Q(C) = Q(C \vee T) + Q(C \vee \neg T) - Q(D)$$

Ví dụ, xét cơ sở dữ liệu thống kê liệt kê các vụ tai nạn ô tô xảy ra dưới đây:

(Trong đó: Customer Name – Tên chủ sở hữu xe, Customer Age – Tuổi, Auto Manufacturer – Hãng sản xuất xe ô tô, Color of Automobile – Màu xe, Time – Thời gian xảy ra vụ tai nạn, At Fault- có lỗi hay không, DUI: có lái xe trong tình trạng bị say và các ảnh hưởng khác hay không?)

Customer Name	Customer Age	Auto Manufacturer	Color of Automobile	Time	At Fault	DUI
J. Parks	21	Honda	Blue	1330	0	1
G. Nguyen	18	Audi	White	0300	0	0
P. Lindstrom	35	Honda	Yellow	1700	1	0
C. Coffee	67	Toyota	Blue	1800	1	1
K. Kuhnhausen	35	Chevrolet	Red	1200	0	0
C. Parks	20	Honda	White	0900	0	0
E. Easterly	21	GM	Silver	1230	0	0

P. Lindstrom	35	Honda	Red	0530	0	1
C. Warner	41	Toyota	Red	0400	1	1
C. Jong	53	Chevrolet	Green	0730	0	0
J. Boucher	24	Volkswagen	Gold	2100	1	1
C. Warner	34	Honda	Blue	1100	0	0

Bảng 5.5. Ví dụ SDB về các vụ tai nạn ô tô

Giả sử ta có công thức đặc trưng:

$$C = (\text{Name} = \text{P.Lindstrom}) \wedge (\text{Color} = \text{Yellow})$$

Các câu truy vấn trên C có kết quả như sau :

$$\text{COUNT}(C) = 1$$

$$\text{SUM}(\text{At Fault}, C) = 1$$

$$\text{SUM}(\text{DUI}, C) = 0$$

Trong trường hợp này chỉ có một người tên là P.Lindstrom liên quan đến vụ tai nạn với chiếc xe màu vàng, và ông ta là người có lỗi trong vụ tai nạn đó.

Áp dụng kỹ thuật hạn chế kích cỡ tập truy vấn để giữ kín thông tin này, ta chọn kích cỡ tối thiểu cho một tập truy vấn đối với SDB ở bảng trên là $k = N/6 = 12/6 = 2$. Kích cỡ tối đa cho phép đối với tập truy vấn là $N-k = 12-2 = 10$. Như vậy, các truy vấn của C là các thống kê nhạy cảm không được đưa ra.

Tuy nhiên, người dùng có thể tìm một công thức T sao cho $k \leq |T| \leq N-k$ nghĩa là $2 \leq |T| \leq 10$. Anh ta có thể bắt đầu bằng việc đoán xấp xỉ một nửa số

bản thi trong bảng 5.5 sẽ tương ứng với những người lái xe có tuổi lớn hơn 25 và kiểm tra điều này bằng công thức T sau:

$$T = (\text{Age} < 25)$$

Rõ ràng, $4 \leq \text{COUNT}(T) = 5 \leq 8$. Nó nằm trong phạm vi cần thiết.

Đây là một trình theo dõi khả thi, giờ đây anh ta có thể tính toán giá trị của bất kỳ câu truy vấn nào liên quan đến C. Dùng Q để ký hiệu các câu truy vấn (COUNT, SUM, MEAN,...)

Anh ta bắt đầu tính: $Q(D) = Q(\text{All}) = Q(T) + Q(\neg T)$

- Với câu truy vấn COUNT: $\text{COUNT}(T) + \text{COUNT}(\neg T) = N$.
- Với câu truy vấn SUM: $\text{SUM}(A_i, T) + \text{SUM}(A_i, \neg T) = \text{Sum}(A_i, D)$ là tổng của tất cả các bản ghi của thuộc tính A_i .

Sau đó, anh ta tính $Q(C)$ theo công thức:

$$Q(C) = Q(C \vee T) + Q(C \vee \neg T) - Q(D)$$

Trong trường hợp này, thu được các kết quả như sau :

$$\text{COUNT}(D) = \text{COUNT}(\text{Age} < 25) + \text{COUNT}(\text{Age} \geq 25) = 5 + 7 = 12$$

$$\begin{aligned} \text{COUNT}(C) &= \text{COUNT}(C \vee \text{Age} < 25) + \text{COUNT}(C \vee \text{Age} \geq 25) - \text{COUNT} \\ (D) &= 6 + 7 - 12 = 1 \end{aligned}$$

$$\text{SUM}(\text{At fault}, D) = \text{SUM}(\text{At fault}, \text{Age} < 25) + \text{SUM}(\text{At fault}, \text{Age} \geq 25)$$

$$= 1 + 3 = 4$$

$$\text{SUM}(\text{At fault}, C) = \text{SUM}(\text{At Fault}, (C \vee \text{Age} < 25))$$

$$+ \text{SUM}(\text{At Fault}, (C \vee \text{Age} \geq 25))$$

$$- \text{SUM}(\text{At Fault}, D)$$

$$= 2 + 3 - 4 = 1$$

Như vậy, với có công thức đặc trưng $C = (\text{Name} = \text{P.Lindstrom}) \wedge (\text{Color} = \text{Yellow})$ có kích cỡ bằng 1. Nếu áp dụng kỹ thuật hạn chế kích cỡ tập truy vấn như ở trên thì, các câu truy vấn như: COUNT(C) hay SUM(At Fault, C)

sẽ không được đưa ra cho người dùng đó. Nhưng nếu áp dụng tấn công Trình theo dõi này, ta có thể được các thông kê trên C.

Có thể định nghĩa các kiểu trình theo dõi phức tạp hơn, chẳng hạn các trình theo dõi chung (*general tracker*), trình theo dõi kép (*double tracker*) và trình theo dõi kết hợp (*union tracker*), chúng hoạt động hiệu quả mặc dù tham số k của kiểm soát kích cỡ tập truy vấn có giá trị gần bằng $N/2$. Hơn nữa, các trình theo dõi chung và kép có thuận lợi là không yêu cầu kiến thức cơ bản về một cá nhân xác định được biểu diễn trong SDB, còn trong các trình theo dõi cá nhân thì có yêu cầu.

5.5.2.1.2 Tấn công hệ tuyến tính

Một kiểu tấn công khác có thể được sử dụng để chống lại kiểm soát kích cỡ tập truy vấn, đó là tấn công hệ tuyến tính. Yêu cầu của tấn công này là giải một hệ các phương trình có dạng:

$$HX = Q$$

Cụ thể như sau:

$$\lambda_{1,1}x_1 + \lambda_{1,2}x_2 + \dots + \lambda_{1,n}x_N = q_1$$

$$\lambda_{2,1}x_1 + \lambda_{2,2}x_2 + \dots + \lambda_{2,n}x_N = q_2$$

.

.

$$\lambda_{k,1}x_1 + \lambda_{k,2}x_2 + \dots + \lambda_{k,n}x_N = q_k$$

Ma trận hệ số H có dạng sau :

$$H = \begin{vmatrix} \lambda_{1,1} & \lambda_{1,2} & \cdot & \cdot & \cdot & \cdot & \lambda_{1,n} \\ \lambda_{2,1} & \lambda_{2,2} & \cdot & \cdot & \cdot & \cdot & \lambda_{2,n} \\ \vdots & \vdots & \ddots & & & & \vdots \\ \lambda_{k,1} & \lambda_{k,2} & \cdot & \cdot & \cdot & \cdot & \lambda_{k,n} \end{vmatrix}$$

Trong đó, $X = (x_1, \dots, x_N)$ là vector của các bản ghi trong SDB và x_i ($1 \leq i \leq N$) là từng bản thi trong cơ sở dữ liệu SDB, $Q = (q_1, \dots, q_k)$ là vector của các thông kê đưa ra, và H là ma trận truy vấn, chẳng hạn như ma trận có kích cỡ $k \times N$ ở trên, trong đó phần tử $H[i,j] = 1$ nếu bản ghi $x_j \in X(C_i)$ - tức là bản ghi x_j của SDB thoả mãn công thức đặc trưng C_i , và bằng 0 nếu ngược lại, trong đó $X(C_i)$, $i = 1, \dots, k$ là một tập truy vấn của thông kê q_i .

Ví dụ, giả thiết trong Employee SDB,

Thông kê $q_1 = \text{Sum } (\text{Sex} = F \wedge \text{Dept-Code} = \text{Dept3} \wedge \text{Birth-Year} = 1968, \text{Salary})$, yêu cầu lương của các nữ nhân công làm việc trong bộ phận "Dept3", sinh năm 1968, trả lại giá trị 33.

Thông kê $q_2 = \text{Sum } ((\text{Sex} = F \vee \text{Sex} = M) \wedge \text{Dept-Code} = \text{Dept3} \wedge \text{Birth-Year} = 1968, \text{Salary})$, yêu cầu tổng số lương của các nhân công sinh năm 1968 trong bộ phận "Dept3", có thể là nam hoặc nữ, trả lại giá trị 37. Hệ tuyến tính có thể như sau:

$$\begin{cases} q1 = \text{Sum } (\text{Sex} = F \wedge \text{Dept-Code} = \text{Dept3} \wedge \text{Birth-Year} = 1968, \text{Salary}) \\ q2 = \text{Sum } ((\text{Sex} = F \vee \text{Sex} = M) \wedge \text{Dept-Code} = \text{Dept3} \wedge \text{Birth-Year} = 1968, \text{Salary}) \end{cases}$$

$$\begin{cases} x1 + x3 + x4 + x6 + x7 + x8 + x9 = 33 \\ x1 + x3 + x4 + x5 + x6 + x7 + x8 + x9 = 37 \end{cases}$$

$$\text{Ta có ma trận } H_{2 \times 9} = \begin{vmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{vmatrix}$$

Trong đó, các bản ghi thỏa mãn (bao gồm trong mỗi thông kê) được liệt kê.

Giả thiết rằng, thông kê:

$\text{Sum}(\text{Sex} = M \wedge \text{Dept-Code} = \text{Dept3} \wedge \text{Birth-Year} = 1968, \text{Salary})$ là nhạy cảm, tập truy vấn của nó có kích cỡ là 1 và nó chỉ ra lương của nhân công Donald. Sau đó thống kê này bị hạn chế bởi kiểm soát kích cỡ tập truy vấn. Tuy nhiên, nếu người dùng sử dụng hệ tuyến tính trên, lương của Donald có thể được tính toán qua x_5 : $x_5 = q_2 - q_1 = 4$.

Một ví dụ khác về tấn công hệ tuyến tính:

Ta xét một cơ sở dữ liệu về công nhân khác như sau:

RowID	Name	Position	Section	Age	Gender	Salary
R1	Adam	Officer	Marketing	29	M	48500
R2	Natasha	Manager	Operations	33	F	62000
R3	Sarah	Officer	Operations	27	F	51000
R4	Smith	Clark	Marketing	24	M	42000
R5	Liz	Officer	Operations	24	F	38500

Bảng 5.6. Một ví dụ khác về SDB công nhân

Ta có: 5 bản ghi $X = (x_1, x_2, x_3, x_4, x_5)$

Với 2 câu truy vấn thống kê sau :

$q_1 = \text{Sum}(\text{Position} = \text{officer} \wedge \text{Gender} = F, \text{Salary})$ gồm 2 bản ghi x_3, x_5 người dùng có thể tính bằng $\text{Count}(\text{Position} = \text{officer}, \text{Gender} = F, \text{Salary}) = 2$.

$q_2 = \text{Sum}(\text{Gender} = F, \text{Salary})$ gồm 3 bản ghi x_2, x_3, x_5 người dùng cũng tính được bằng $\text{Count}(\text{Position} = \text{officer} \wedge \text{Gender} = F, \text{Salary}) = 3$

Ta thấy rằng câu truy vấn:

$q_3 = \text{Sum}(\text{Position} = \text{Manager}, \text{Gender} = F, \text{Salary})$ là câu truy vấn nhạy cảm vì kích cỡ của nó bằng 1, liên quan đến bản ghi x_2 . Do đó nó sẽ không được đưa ra nếu hệ thống sử dụng Kiểm soát kích cỡ tập truy vấn. Tuy nhiên người dùng có thể tính được thống kê này bằng hệ tuyến tính gồm 2 phương trình sau:

$$q_1 = x_3 + x_5 = 89.500$$

$$q_2 = x_2 + x_3 + x_5 = 151.500$$

Ma trận truy vấn :

$$H_{2 \times 5} = \begin{vmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \end{vmatrix}$$

$$\text{Ta có } x_2 = 151500 - 89500 = 62000$$

Như vậy, nếu người dùng biết rằng chỉ có 1 nữ nhân công ở vị trí Manager thì họ đã suy diễn được lương của nhân công này.

5.5.2.2 Kiểm soát kích cỡ tập truy vấn mở rộng

Kiểm soát kích cỡ tập truy vấn mở rộng

Như đã trình bày ở trên, cơ chế cơ bản của kiểm soát kích cỡ tập truy vấn không đủ đảm bảo chống suy diễn, do vẫn có thể tính toán được các thống kê nhạy cảm, thông qua một tập các thống kê được phép.

Người ta đã cải tiến kiểm soát này. Từ việc phân tích các tấn công dựa vào trình theo dõi và dựa vào các hệ tuyến tính, chúng ta có thể nhận ra rằng các công thức đặc trưng (được sử dụng trong các tấn công này) thường có quan hệ nào đó với các công thức đặc trưng khác. Ta có thể mở rộng kiểm soát kích cỡ tập truy vấn bằng cách tăng số lượng các tập truy vấn cần được kiểm soát để quyết định khi nào thì một thống kê được phép và khi nào có thể đưa ra thống kê đó cho người sử dụng.

Chính xác hơn, với một công thức đặc trưng C cho trước, chúng ta có thể định nghĩa các *tập truy vấn ngầm định*. Cụ thể là các tập truy vấn ngầm định sẽ được định dạng trực tiếp bằng cách kết hợp các thuộc tính của C . Sau đó, các tập truy vấn này sẽ được kiểm soát để quyết định khi nào thì có thể trả lời câu truy vấn của người dùng.

Cho trước một thống kê bậc m có dạng như sau:

$$q(A_1 = a_1 \wedge A_2 = a_2 \wedge \dots \wedge A_m = a_m)$$

hoặc có dạng:

$$q(A_1 = a_1 \vee A_2 = a_2 \vee \dots \vee A_m = a_m)$$

Khi đó, tồn tại $2^m = C_m^0 + C_m^1 + C_m^2 + \dots + C_m^{m-1}$ tập truy vấn ngầm định, tương ứng với các thông kê sau đây:

$$q(A_1 = a_1 \wedge A_2 = a_2 \wedge \dots \wedge A_m = a_m)$$

$$q(A_1 = a_1 \wedge A_2 = a_2 \wedge \dots \wedge \neg A_m = a_m)$$

...

$$q(A_1 = a_1 \wedge \neg A_2 = a_2 \wedge \dots \wedge A_m = a_m)$$

$$q(\neg A_1 = a_1 \wedge A_2 = a_2 \wedge \dots \wedge A_m = a_m)$$

...

$$q(\neg A_1 = a_1 \vee \neg A_2 = a_2 \vee \dots \vee \neg A_m = a_m)$$

Ta dễ thấy rằng, một thông kê bậc m được phép (thông kê hợp lệ) nếu tất cả 2^m tập truy vấn ngầm định của nó tuân theo kiểm soát kích cỡ tập truy vấn, có nghĩa là nếu chúng nằm trong khoảng $[k, N-k]$. Trong thực tế, nếu kiểm tra hết 2^m tập truy vấn, ta sẽ kiểm tra được xem mỗi tập truy vấn ngầm định đó có ít nhất k bản ghi hay không và để đảm bảo rằng không tồn tại tập truy vấn nào có nhiều hơn $N-k$ bản ghi. Bằng cách mở rộng kiểm soát kích cỡ tập truy vấn với các tập truy vấn ngầm định, chúng ta có thể ngăn chặn được các tấn công (như đã được trình bày trong mục trước).

Ví dụ, có đặc trưng:

$$C = (Sex = F \wedge Dept-Code = Dept1 \wedge Birth-Year = 1951).$$

Biến đổi $q(C)$ như sau:

$$q(C) = q(Sex = F) - q(Sex = F \wedge \neg(Dept-Code = Dept1 \wedge Birth-Year = 1951))$$

$$q(C) = q(Sex = F) - q(Sex = F \wedge (\neg Dept-Code = Dept1 \vee \neg Birth-Year = 1951)).$$

Như vậy, với tập truy vấn tương ứng với đặc trưng: $T = (Sex = F \wedge (\neg Dept-Code = Dept1 \vee \neg Birth-Year = 1951))$, người dùng có thể suy diễn ra thông kê của C. Tuy nhiên, nếu áp dụng kiểm soát này, T cũng được kiểm tra vì nó là một tập truy vấn ngầm định của C và do đó ta có thể ngăn chặn được tấn công.

Tuy nhiên cần lưu ý rằng, vì số lượng các tập truy vấn ngầm định được kiểm tra tăng theo hàm mũ với bậc của thông kê, do đó việc tăng số lượng m thuộc tính làm cho chi phí kiểm soát càng đắt và không thực tế với các thông kê bậc cao. Nghĩa là với thông kê bậc m lớn thì kiểm soát này không hiệu quả.

Bên cạnh đó, các thông kê nhạy cảm có thể được đưa ra qua nhiều cách, cụ thể là bằng cách phân tích tất cả các tập truy vấn ngầm định của chúng, như trong ví dụ sau đây.

Chúng ta xét 2 thuộc tính A_i và A_j của các thực thể trong cơ sở dữ liệu. Nếu thuộc tính A_i có n giá trị (a_{i1}, \dots, a_{in}) và A_j có p giá trị (a_{j1}, \dots, a_{jp}), các bản ghi của SDB được phân hoạch thành $(n \times p)$ nhóm nhỏ, mỗi nhóm được thể hiện bởi các đặc trưng kiểu $(a_{it} \wedge a_{jq})$, với $t = 1, \dots, n$ và $q = 1, \dots, p$. Chúng ta giả thiết rằng $q(a_{i1} \wedge a_{j1})$ là một thông kê nhạy cảm và tất cả các thông kê còn lại $q(a_{it} \wedge a_{jr})$, với $t = 2, \dots, n$ và $r = 2, \dots, p$ không phải là các thông kê nhạy cảm. Kiểm soát kích cỡ tập truy vấn mở rộng sẽ kiểm tra các tập truy vấn ngầm định của $q(a_1 \wedge b_1)$ và ngăn chặn các câu truy vấn không hợp pháp dựa vào các tập truy vấn này.

Như vậy với truy vấn $q(a_{i1} \wedge a_{j1})$, thì các tập truy vấn ngầm định là $2^2 = 4$, bao gồm: $q(a_{i1} \wedge a_{j1})$, $q(a_{i1} \wedge \neg a_{j1})$, $q(\neg a_{i1} \wedge a_{j1})$, $q(\neg a_{i1} \wedge \neg a_{j1})$.

Do vậy có thể tránh được các tấn công như sau:

$$q(a_{i1} \wedge a_{j1}) = q(a_{j1}) - q(a_{j1} \wedge \neg a_{i1})$$

Tuy nhiên, có thể tính toán được thông kê nhạy cảm $q(a_{i1} \wedge a_{j1})$, bằng cách sử dụng các thông kê cho phép, chẳng hạn như:

$$q(a_{i1} \wedge a_{j1}) = q(a_{j1}) - [q(a_{i2} \wedge a_{j1}) + \dots + q(a_{in} \wedge a_{j1})]$$

Trong đó, người sử dụng có thể sử dụng các thông kê không bao gồm các giá trị a_{i1} và a_{j1} cùng thời điểm.

Để ngăn chặn kiểu tấn công này, chúng ta nên tăng số lượng các tập truy vấn cần kiểm tra.

Như vậy, kiểm soát kích cỡ tập truy vấn mở rộng vẫn có thể bị tấn công nếu người dùng sử dụng các truy vấn không có công thức đặc trưng nằm trong tập truy vấn ngầm định của công thức đặc trưng trong câu truy vấn ban đầu (câu truy vấn ban đầu sẽ bị hạn chế bởi kiểm soát kích cỡ tập truy vấn).

5.5.2.3 Kiểm soát chòng lấp tập truy vấn (query-set overlap control)

Một cách khác để cải tiến kiểm soát kích cỡ tập truy vấn là tác động vào số lượng các bản ghi chung mà các đặc trưng của nó được sử dụng trong các tấn công dựa vào trình theo dõi và các tấn công hệ tuyến tính. Chúng ta cũng có thể xem trong các ví dụ trước, các tấn công này được xác định rõ thông qua các thông kê liên tiếp, các đặc trưng của chúng có số lượng các bản ghi chung rất cao. Ý tưởng của kỹ thuật kiểm soát chòng lấp tập truy vấn là các câu truy vấn liên tiếp phải được kiểm tra, dựa vào số lượng các bản ghi chung của chúng, không cho phép chúng có số lượng bản ghi chung lớn hơn so với ngưỡng cho trước. Chính xác hơn, kỹ thuật *kiểm soát chòng lấp tập truy vấn* cho phép một thông kê yêu cầu $q(C)$ chỉ khi:

$$|X(C) \cap X(D)| \leq \alpha, \alpha > 0$$

Có nghĩa là chỉ khi số lượng các bản ghi (giữa tập truy vấn của $q(C)$ và tập truy vấn của tất cả các thông kê $q(D)$) đã được đưa ra phải nhỏ hơn hoặc ngang bằng với α (α là ngưỡng của hệ thống, chỉ ra số lượng các bản ghi chung tối đa được phép cho các tập truy vấn của 2 câu truy vấn liên tiếp).

Với k là tham số của kiểm soát kích cỡ tập truy vấn, α là giới hạn chòng lấp. theo Dobkin (1979) cận dưới e cho số lượng các truy vấn cần thiết để làm lộ SDB là $e = 1 + (k-1)/\alpha$. Thậm chí nếu kiểm soát này thành công trong việc ngăn chặn các tấn công (chẳng hạn như các tấn công dựa vào hệ tuyến tính đã được trình bày ở trên), nó vẫn có thể bị phá vỡ do các tấn công được thiết kế

theo dạng chuỗi các câu truy vấn liên tiếp, vẫn qua được kiểm soát chòng lấp, chẳng hạn như trong ví dụ sau, với s là hệ tuyến tính được sử dụng để suy luận ra x_3 bằng cách sử dụng 3 câu truy vấn, với $\alpha = 2$:

$$q_1 = x_1 + x_2$$

$$q_2 = x_2 + x_3$$

$$q_3 = x_1 + x_3$$

$$\text{Thì } x_3 = (q_3 + q_2 - q_1)/2, \text{ rõ ràng là số bản ghi chung ở đây bằng } 1 < \alpha = 2$$

Kiểm soát này trở nên kém hiệu quả, do nó yêu cầu nhiều so sánh (mỗi câu truy vấn mới phải được kiểm tra, dựa vào từng câu truy vấn đưa ra trước đó, để kiểm tra điều kiện chòng lấp trên các tập truy vấn tương ứng của chúng). Hơn nữa nó có thể hạn chế các khả năng hữu ích của SDB, bởi vì các thông kê (chứa trong một tập và các tập con của nó) không thể được đưa ra do số lượng các bản thi chung của các truy vấn này $> \alpha$ (ví dụ, các câu truy vấn trên tập nữ nhân công và tập nữ nhân công có năm sinh lớn hơn một năm sinh cụ thể nào đó bị hạn chế). Thời gian tăng thêm khi xử lý từng câu truy vấn mới với điều kiện chòng lấp là $O(N)$, trong đó N là số lượng các bản ghi của SDB. Hơn nữa, kiểm soát có thể bị phá vỡ dễ dàng hơn, do các tấn công dựa vào hệ tuyến tính và do sử dụng các câu truy vấn xác định khoá.

5.5.2.4 Kiểm toán

Kiểm soát chòng lấp câu truy vấn làm nảy sinh nhiều kiểu kiểm soát nhằm phát hiện các đe doạ suy diễn, bằng cách so sánh từng câu truy vấn thống kê mới với các câu truy vấn được đưa ra trước đó. Các kiểm soát dựa vào kiểm toán đã được phát triển theo hướng này, bằng cách lưu giữ nhật ký và hồ sơ người dùng để ghi lại lịch sử của các câu truy vấn do người dùng yêu cầu, đồng thời đối chiếu từng câu truy vấn mới với các câu truy vấn đã được đưa ra trước đó, để xem câu truy vấn mới có thể dẫn đến lộ SDB hay không.

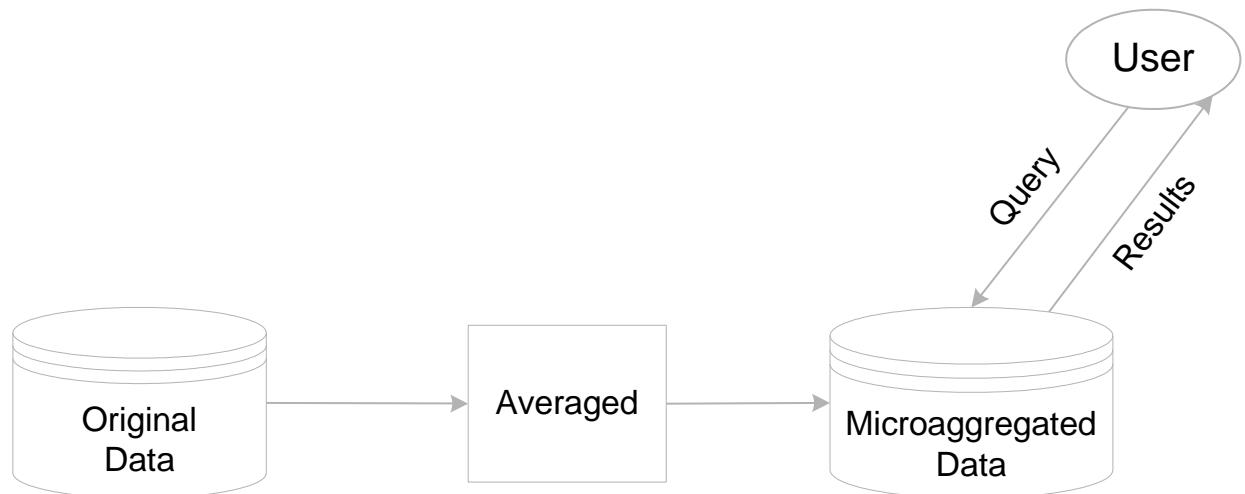
Ví dụ, một hướng tiếp cận dựa vào việc sử dụng các *cảm biến*. *Cảm biến* là một chức năng được sử dụng để xác định xem các câu truy vấn của người

sử dụng có thể được trả lời hay không, dựa trên câu truy vấn được đưa ra và kiến thức thực tế của người sử dụng, được lưu giữ trong một *cơ sở thông tin giả định*. Nó lưu giữ dấu vết về những gì mà người sử dụng biết được về các nội dung của SDB và nó cập nhật liên tục các thông tin có được từ mỗi câu truy vấn đã được đưa ra.

Hệ thống Audit Expert – chuyên gia kiểm toán là hệ thống có thể phát hiện các suy diễn có thể, bằng cách đối chiếu câu truy vấn mới với các câu truy vấn đã được đưa ra trước đó, sử dụng biểu diễn phù hợp các câu truy vấn trong bộ nhớ.

5.5.2.5 Gộp (Microaggregation)

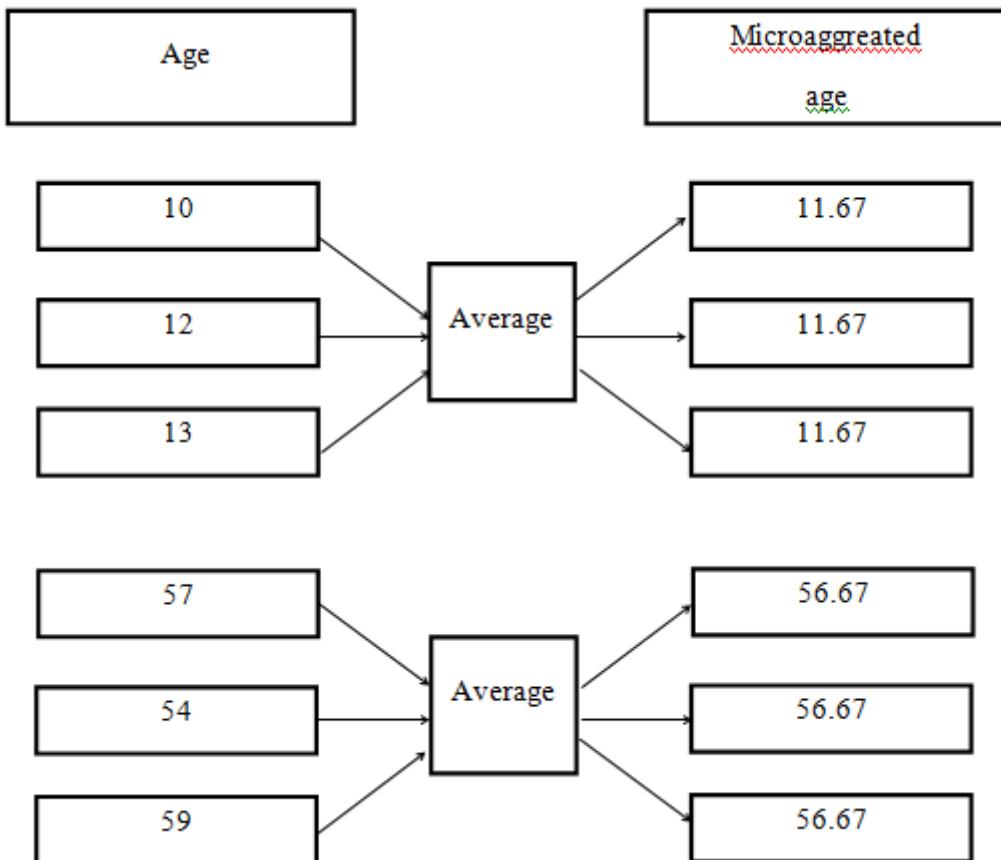
Trong kỹ thuật Gộp, các dữ liệu riêng (thô) sẽ được nhóm lại thành một khối nhỏ trước khi đưa ra. Giá trị trung bình của nhóm gộp sẽ thay thế cho mỗi giá trị riêng của dữ liệu được gộp. Dữ liệu giống nhau sẽ được nhóm lại để đảm bảo tính chính xác của dữ liệu. Kỹ thuật này giúp ngăn chặn khám phá dữ liệu riêng.



Hình 5.15. Kỹ thuật gộp

Cục thống kê nông nghiệp quốc gia (NASS) của Mỹ công bố dữ liệu về các nông trường, trang trại. Để bảo vệ chống lại sự khám phá dữ liệu, dữ liệu chỉ được đưa ra ở mức country - vùng. Các nông trại ở mỗi vùng sẽ được gộp để bảo vệ tính riêng tư và tránh bị khám phá.

Ví dụ về kỹ thuật gộp được mô tả ở hình dưới đây:



Hình 5.16. Ví dụ về kỹ thuật gộp

5.5.2.6 Giấu ô

Kỹ thuật này được thiết kế cho các SDB (đưa ra các thống kê trong bảng 2- chiều của các thống kê vĩ mô, ví dụ các thống kê dân số).

Kỹ thuật giấu ô như sau: trong các bảng, giấu đi tất cả các ô tương ứng với các thống kê nhạy cảm (*Giấu*) và các ô tương ứng với các thống kê có thể gián tiếp dẫn đến tình trạng khám phá các thống kê nhạy cảm (*Giấu bổ sung*).

Cần có một tiêu chuẩn để xác định khi nào một thống kê (một ô) là nhạy cảm, hoặc có thể gây ra tình trạng lộ các thống kê nhạy cảm.

Nói chung, với các thống kê *Count*, tiêu chuẩn nhạy cảm là kích cỡ tối thiểu của tập truy vấn (ví dụ, các tập truy vấn có kích cỡ bằng 1). Các thống kê (các ô) có giá trị nhỏ hơn hoặc bằng kích cỡ tối thiểu này đều được giấu đi. Trong trường hợp các ô có kích cỡ bằng 1, người ta đã đề xuất một kỹ thuật để đối phó như sau: kết hợp một ô có kích cỡ bằng 1 với một ô có kích cỡ lớn

hơn 1. Tuy nhiên, hướng tiếp cận này có thể gây ra tình trạng mất mát thông tin không nhỏ (khoảng 50%), vì vậy trên các SDB mục đích chung, nó kém hiệu quả hơn.

Với các thống kê *Sum*, tiêu chuẩn nhạy cảm được sử dụng là quy tắc «*đáp ứng n, trội k%* ». Theo tiêu chuẩn này, một thống kê là nhạy cảm nếu các giá trị thuộc tính của n hoặc ít hơn n thực thể (bản ghi) tạo thành $k\%$ hoặc lớn hơn $k\%$ trong toàn bộ thống kê *Sum* đó, thì ô này cần được giấu đi. Các tham số n và k được giữ bí mật và do DBA xác định.

Xét thống kê $q(C) = \text{sum}(C, A_j)$ và các tham số k, n trong tiêu chuẩn «*đáp ứng n, trội k%* ». Thống kê $q(C)$ là nhạy cảm nếu tổng thành phần của nó (chứa n giá trị trội) chiếm cao hơn $k\%$ của tổng $q(C)$.

Mô hình hóa kỹ thuật giấu ô với thống kê *Sum* như sau :

Giả sử thống kê $q(C)$ được tính trên các giá trị của thuộc tính A_j bao gồm M bản ghi như sau:

$$q(C) = a_{j1} + a_{j2} \dots + a_{jn} + \dots + a_{jM}$$

Ta giả sử tổng $a_{j1} + a_{j2} \dots + a_{jn}$ là tổng của n giá trị trội nhất với:

$$a_{j1} \geq a_{j2} \geq \dots \geq a_{jn} \geq \dots \geq a_{jN}. \text{Đặt } d = a_{j1} + a_{j2} \dots + a_{jn} \text{ và } q = q(C).$$

Khi đó, thống kê $q(C)$ là nhạy cảm (cần được giấu) nếu $d > (k/100)q$.

Ví dụ

Giả sử $n = 1$ và $k = 90\%$ và xét bảng hai chiều của thống kê vĩ mô ở bảng 1 sau, báo cáo về tổng lương của các nam và nữ công nhân trong phòng ‘Dept1’, ‘Dept2’, ‘Dept3’.

Sex	Department			Sum Salary
	Dept1	Dept2	Dept3	
M	135	80	50	219
F	120	360	100	580
Sum Salary	255	440	150	732

Bảng 5.7. SDB vĩ mô về công nhân

Nếu chỉ có 1 công nhân nam làm ở phòng ‘Dept3’ thì ta có :

$\text{Count}(\text{Department} = \text{Dept3} \wedge \text{Sex} = M) = 1$, và do đó ô (1,3) là ô nhạy cảm cần phải giấu đi vì lương của công nhân này tạo thành 100% của toàn bộ tổng lương tại ô đó (với $n=1$ trội $100\% > 90\%$). Cũng như vậy, ô (2,3) cần phải giấu đi (*giấu bớt* *sung*) vì nếu lấy tổng của cột 3 trừ đi tổng ở ô (2,3) sẽ tìm được tổng của ô (1,3).

Do đó từ bảng trên, áp dụng kỹ thuật giấu ô ta được bảng sau:

Sex	Department			Sum Salary
	Dept1	Dept2	Dept3	
M	135	80	-	219
F	120	360	-	513
Sum salary	255	440	37	732

Bảng 5.8. SDB vĩ mô về công nhân được giấu ô

Tuy nhiên, trong bảng 5.8 ta vẫn cần phải giấu bớt sung, bởi vì các ô được giấu vẫn có thể bị suy diễn ra bằng các ô khác trong bảng. Thật vậy, trong

bảng trên ta vẫn có thể suy ra giá trị trong các ô (1,3) và (2,3) bằng cách lấy tổng hàng trừ đi các ô còn lại (vì các ô này không nhạy cảm nên có thể lấy được).

Như vậy, quy tắc *giấu bỏ sung* là: nếu có một ô nhạy cảm thì phải giấu đi ít nhất một ô cùng hàng và một ô cùng cột với ô đó. Ta thu được bảng tiếp theo:

Sex	Department			Sum Salary
	Dept1	Dept2	Dept3	
M	–	80	–	219
F	–	360	–	513
Sum salary	255	440	37	732

Bảng 5.9. SDB vĩ mô về công nhân được giấu bỏ sung

5.5.2.7 Kỹ thuật kết hợp

Trong mục này chúng ta sẽ xem xét kỹ thuật *giấu từng phần* (*partial suppression*), đây là kỹ thuật hạn chế bằng cách kết hợp các khái niệm về phân hoạch, kích cỡ tập truy vấn và giấu ô, tạo thành một giải pháp tích hợp. Mục đích là đưa ra số lượng các thông kê lớn hơn so với các kỹ thuật hạn chế đó.

Kỹ thuật giấu từng phần hoạt động như sau:

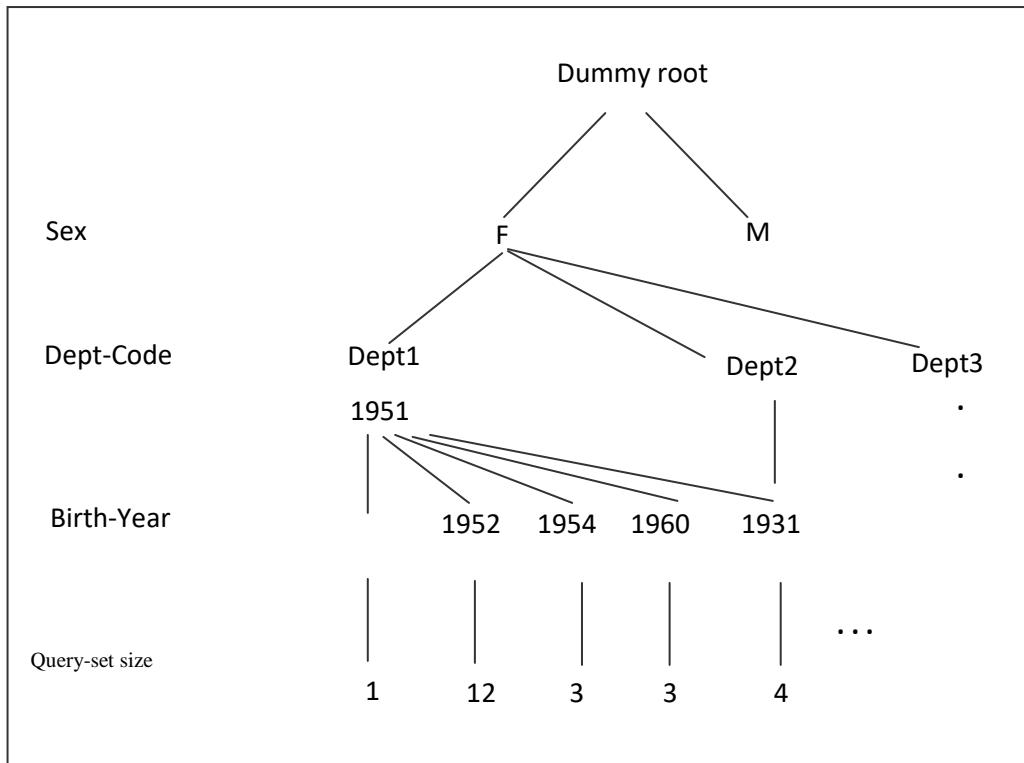
- Bước 1: Phân hoạch các bản ghi SDB thành một số các tập con riêng biệt (các tập truy vấn), dựa vào các giá trị của các thuộc tính được đánh chỉ số (index) và được chọn lựa (trong lược đồ quan hệ, một thuộc tính được đánh chỉ số nếu một chỉ số được gắn liền với nó để có thể truy nhập nhanh vào các bản ghi có trong file, kỹ thuật này chỉ sử dụng các thuộc tính đã được đánh chỉ số của lược đồ SDB để phân hoạch các bản ghi).

- Bước 2: Nhận dạng các tập truy vấn có kích cỡ không thoả mãn "kiểm soát dựa vào kích cỡ tập truy vấn".
- Bước 3: Đánh giá một số điều kiện đã được định nghĩa trước để quyết định xem tập truy vấn nào cần được giấu đi.

Giả sử $q(C)$ là một thóng kê yêu cầu. Tập truy vấn của thóng kê này, $X(C)$ được gọi là *tập truy vấn cơ bản (BQS)* nếu nó tương ứng với một tập con là kết quả của kỹ thuật phân hoạch. Như vậy BQS phải chứa đầy đủ các thuộc tính của SDB. BQS được gọi là *tập truy vấn không đầy đủ (IQS)* nếu kích cỡ của nó không thoả mãn kiểm soát kích cỡ tập truy vấn (có nghĩa là $|X(C)| < k$) và là *tập truy vấn đủ (SQS)* nếu ngược lại.

Hình 5.17 minh họa một ví dụ về cây phân hoạch. Trong đó, các bản ghi của Employee SDB được phân hoạch, tuỳ thuộc vào các thuộc tính được đánh chỉ số là *Sex*, *Dept-Code* và *Birth-Year*.

Tập truy vấn tương ứng với đặc trưng $C = (Sex = F \wedge Dept-Code = Dept1 \wedge Birth-Year = 1951)$ là một IQS, khi kích cỡ của nó là 1 (chú ý rằng, bản thân tập truy vấn tương ứng với C đã là một BQS) . Tập truy vấn tương ứng với đặc trưng $C = (Sex = F \wedge Dept-Code = Dept1 \wedge Birth-Year = 1952)$ là một SQS.



Hình 5.17 Ví dụ về cây phân hoạch

Sau khi phân hoạch các bản ghi của SDB và nhận dạng các IQS, cần đánh giá một số điều kiện để quyết định xem cần giấu đi những tập truy vấn nào. Các điều kiện đó như sau: (chú ý m là số thuộc tính của SDB)

(1) Nếu đặc trưng C có m thuộc tính được đánh chỉ số, hoặc $(m-1)$ thuộc tính được đánh chỉ số (không bao gồm thuộc tính thứ m) được kết nối bởi toán tử \wedge , thì một IQS cần được giấu đi.

Ví dụ, chúng ta xét một IQS có kích cỡ 1, tương ứng với đặc trưng $C = (Sex = F \wedge Dept-Code = Dept1 \wedge Birth-Year = 1951)$. Theo điều kiện trên, IQS này luôn luôn phải được giấu đi, bằng cách sử dụng kỹ thuật giấu ô và giấu từng phần. Nếu chúng ta đưa ra câu truy vấn: $q(C_1) = Count(Sex = F \wedge Dept-Code = Dept1)$, bằng cách sử dụng kỹ thuật giấu ô, kết quả đưa ra cho người dùng là 19 (hình 4.6) vì trong kỹ thuật giấu ô, ta chỉ dấu đi ô tương ứng với kích cỡ bằng 1, nhưng giá trị đếm hay tổng toàn bộ vẫn được đưa ra chính xác. Tuy nhiên, nếu sử dụng kỹ thuật giấu từng phần, kết quả được trả lại cho người dùng là 18, mặc dù

không chính xác, nhưng kỹ thuật này cho phép đưa ra nhiều thông kê hơn. Thật vậy, xét câu truy vấn sau đây:

$q(C_2) = \text{Count}(Sex = F \wedge Dept-Code = DeptI \wedge \neg Birth-Year = 1951)$,
kết quả bằng 18.

Nếu ta sử dụng kỹ thuật giấu ô, thì thông kê này cần được giấu đi, vì khi kết hợp với $q(C_1)$, nó làm lộ kích cỡ của IQS nhạy cảm ở trên ($q(C)=1$) (nghĩa là, $q(C_1) - q(C_2) = 19-18 = q(C) = 1$). Nhưng nếu ta sử dụng kỹ thuật giấu từng phần, thông kê $q(C_2)$ có thể được đưa ra, vì không thể sử dụng câu trả lời dành cho $q(C_1)$ để suy diễn (nghĩa là, $q(C_1) - q(C_2) = 18-18 = 0$).

(2) Nếu C có $(m-1)$ thuộc tính được đánh chỉ số và thuộc tính thứ m là một trong $(m-1)$ thuộc tính này, thì một IQS có thể được đưa ra nếu cả hai điều kiện sau thoả mãn:

- (a) Có một hoặc nhiều tập truy vấn cơ bản (BQS)
- (b) Các BQS đó và IQS có cùng giá trị cho thuộc tính thứ m .

Chúng ta xét câu truy vấn $q(C_3) = \text{Count} (Sex = F \wedge Birth-Year = 1951)$ (kết quả bằng 1) có chứa hai thuộc tính được đánh chỉ số. Khi tham chiếu vào cây phân hoạch trong hình 5.17, chúng ta biết được số lượng các thuộc tính được đánh chỉ số là 3 ($m=3$) và thuộc tính thứ 3 là *Birth-Year*.

Bây giờ ta kiểm tra hai điều kiện (a) và (b). Ta thấy rằng, đặc trưng C_3 xác định một IQS là tập truy vấn của đặc trưng sau: $Sex = F \wedge Dept-Code = DeptI \wedge Birth-Year = 1951$. Đồng thời C_3 cũng xác định một tập truy vấn cơ bản (BQS), cũng chính là IQS trên (nhắc lại, BQS phải chứa cả m thuộc tính của SDB). Đồng thời cả IQS và BQS đều có cùng giá trị ở thuộc tính thứ $m=3$, đó là *Birth-Year = 1951*.

Như vậy, cả hai điều kiện (a) và (b) đều thoả mãn. Do đó, IQS trên không cần phải giấu.

(Chú ý, với từng thống kê, ta xác định được các IQS khác nhau, do đó có thể quyết định là giấu hay không giấu IQS đó tuỳ thuộc vào từng yêu cầu thống kê của người dùng).

(3) Nếu C có $1, 2, \dots$, hoặc $(m-1)$ thuộc tính được đánh chỉ số, các điều kiện (1) và (2) được kiểm tra, thì IQS có thể được đưa ra.

Kỹ thuật giấu từng phần là một cố gắng nhằm cải tiến kỹ thuật giấu ô, bằng cách đưa ra một số điều kiện cho phép đưa ra một số thống kê. Hạn chế của nó là thiếu chính xác, vì các thống kê được đưa ra với các câu trả lời không chính xác. Để đảm bảo độ chính xác chấp nhận được, DBA phải định nghĩa một ngưỡng tỉ lệ lỗi và ngưỡng này có thể phụ thuộc vào chiều của SDB, kích cỡ của tập truy vấn và tỷ lệ lỗi mong muốn; Có đưa ra các thống kê có tỷ lệ lỗi lớn hơn ngưỡng, đảm bảo chống suy diễn trên các IQS (đây là các IQS được sử dụng cho việc tính toán các thống kê).

5.5.3 Các kỹ thuật gây nhiễu

Lớp kỹ thuật này ngăn chặn suy diễn, bằng cách đưa ra một số kiểu sửa đổi nào đó trong quá trình xử lý một câu truy vấn thống kê, với mục đích có thể đưa ra nhiều thống kê hơn so với các kỹ thuật dựa vào hạn chế. Việc sửa đổi có thể được áp dụng cho các bản ghi được lưu giữ trong SDB, các bản ghi này được sử dụng khi tính toán các thống kê yêu cầu (*kỹ thuật gây nhiễu dữ liệu*), hoặc được áp dụng cho kết quả chính xác trước khi chuyển nó cho người sử dụng (*kỹ thuật gây nhiễu đầu ra*).

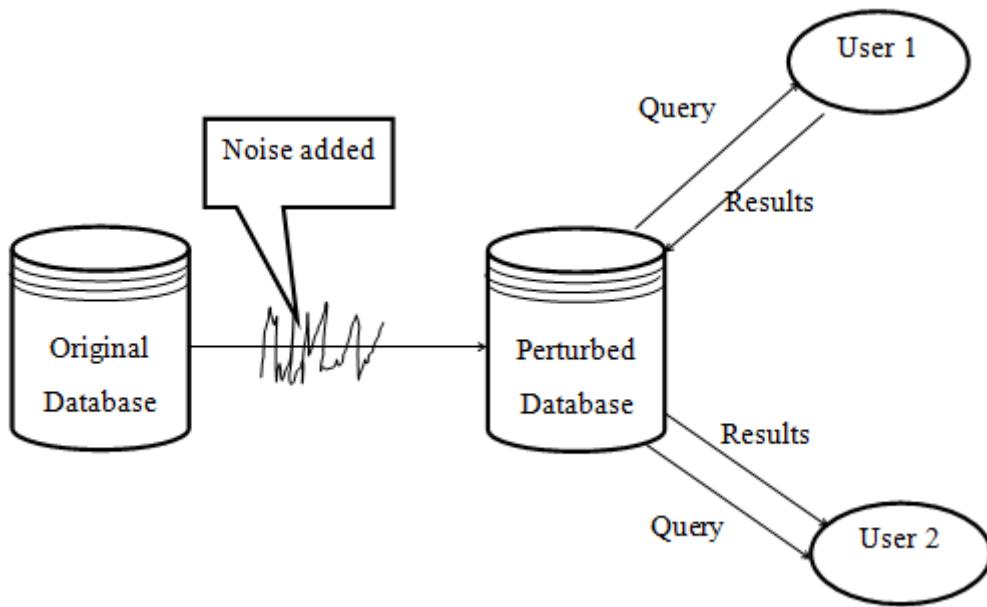
Như vậy, kỹ thuật dựa vào gây nhiễu được phân thành hai loại: kỹ thuật gây nhiễu dữ liệu (Data Perturbation) và kỹ thuật gây nhiễu đầu ra (Output Perturbation).

5.5.3.1 Kỹ thuật gây nhiễu dữ liệu

Các kỹ thuật gây nhiễu dữ liệu tiến hành trên các giá trị của bản ghi (được lưu giữ trong SDB) và sinh ra một dữ liệu mới – “dữ liệu bị sửa đổi”. Dữ liệu này được sử dụng khi tính toán các câu truy vấn thống kê. Việc sửa đổi các nội dung của cơ sở dữ liệu phải tuân theo các giải pháp khác nhau.

Kỹ thuật gây nhiễu dữ liệu bao gồm:

- Chuyển đổi dữ liệu
- Các câu truy vấn mẫu ngẫu nhiên
- Gây nhiễu cố định
- Gây nhiễu dựa vào truy vấn



Hình 5.18. Mô tả kỹ thuật gây nhiễu dữ liệu

5.5.3.1.1 Kỹ thuật chuyển đổi dữ liệu (Data Swapping)

Kỹ thuật này còn được gọi là *chuyển đổi đa chiều* (Schlorer, 1981) tạo ra một SDB sửa đổi, trong đó các thống kê bậc t đầu tiên phải chính xác (thống kê bậc t là một thống kê được tính toán trên t thuộc tính (A_1, \dots, A_t) của các bản ghi SDB), có nghĩa là chúng trùng khớp với các thống kê của SDB gốc, với một tham số t nào đó và các thống kê bậc cao hơn không nhất thiết phải chính xác.

Thực tế, kỹ thuật này có nguồn gốc từ kỹ thuật chuyển đổi các giá trị thuộc tính giữa các bản ghi của SDB gốc, theo cách này SDB thu được sau khi sửa đổi sẽ không có các bản ghi chung với SDB gốc, trong khi đó vẫn đảm bảo chống suy diễn và đảm bảo độ chính xác của các thống kê bậc t đầu tiên được đưa ra.

Kỹ thuật chuyển đổi dữ liệu dựa vào các khái niệm và các thuộc tính (được định nghĩa cho các ma trận) và sau đó được mở rộng cho các SDB, với giả thiết rằng các bản ghi của SDB không được nhận dạng dựa vào khoá và SDB đó phải có kiểu tĩnh (thường là các SDB có mục đích đặc biệt). Sự mở rộng này cho các SDB dựa vào khả năng ánh xạ các bản ghi của một SDB thành một ma trận.

Như vậy kỹ thuật này thường chỉ áp dụng cho các SDB tĩnh.

Giả sử D là một SDB bao gồm một tập N bản ghi, $D = (x_1, \dots, x_N)$ được đặc trưng bởi M thuộc tính A_1, \dots, A_M . Với mỗi thuộc tính A_j đều có c_j giá trị trong tập $M_j = \{a_{j1}, a_{j2}, \dots, a_{jc_j}\}$, ta định nghĩa một hàm f_j ánh xạ từ tập M_j của mỗi thuộc tính vào tập số nguyên không âm $N_0 = \{0, 1, 2, \dots\}$ như sau:

$$f_j: M_j \rightarrow N_0 \mid f_j(a_{jk}) = k - 1, j = 1 \dots M, k = 1, \dots, c_j$$

Giả sử F là một tập các hàm f_j , $F = \{f_j\}$.

Khi đó, $F(D)$ được gọi là *ma trận ảnh* của D trong tập N_0 , đây là ma trận có kích cỡ $N \times M$. Đặt $F(D) = M$, khi đó M được gọi là ma trận biến đổi *d-chiều* (*d-transformable*).

Một ma trận $M (n \times m)$ là *d-transformable* nếu tồn tại ít nhất một ma trận $\tau (n \times m)$ sao cho :

- (i) M và τ là tương đương d-chiều (d-equivalent)
- (ii) M và τ không có hàng nào giống nhau

Ta định nghĩa:

- *Các ma trận tương đương (equivalent matrices)*: hai ma trận A và B được gọi là tương đương nếu tồn tại một hàm hoán vị hàng σ sao cho:

$$\sigma(A) = B \leftrightarrow \sigma^{-1}(B) = A$$

- *Các ma trận tương tự (analogue matrices)*: giả sử A và B là hai ma trận kích cỡ $n \times m$. Giả sử $A^r = (A^{j1}, A^{j2}, \dots, A^{jr})$ và $B^r = (B^{k1}, B^{k2}, \dots, B^{kr})$ là hai ma trận con cỡ $n \times r$, với $r = 1 \dots m$, và $A^r (B^r)$ là một vector cột của $A^r (B^r)$. A^r và B^r là các ma trận con tương tự của A và B nếu:

$$j_i = k_i, \quad i = 1, \dots, r$$

- *Các ma trận d-equivalent* : Giả $d \in \{0, 1, \dots, m\}$. Hai ma trận A và B là *d-equivalent* nếu tất cả các cặp ma trận con tương tự có thể $(n \times d)$ đều tương đương.

Như vậy, với một cặp ma trận biến đổi d-chiều của nhau $\{M, \tau\}$ (cũng có thể gọi là *d-pair*) thì SDB D có M là ma trận ảnh của nó, D cũng có thể được

biến đổi d-chiều, nghĩa là nó có thể được biến đổi thành một SDB mới $D' = F(\tau)$. Với d -pair $\{D, D'\}$ là hai SDB có thể biến đổi d-chiều lẫn nhau, các bản ghi của chúng có nội dung khác nhau, nhưng các thông kê bậc k ($1 \leq k \leq d$) thì giống nhau. Vì vậy D' có thể được dùng thay cho D để đưa ra các thông kê bậc d đầu tiên, mà không làm mất tính chính xác của các thông kê đó, đồng thời có thể ngăn chặn suy diễn.

Trong thực hành thì ta lấy được D' từ D bằng cách chuyển đổi các giá trị thuộc tính giữa các bản ghi của D nhằm bảo vệ các thông kê. Ví dụ, để bảo vệ các thông kê đếm bậc 1, có nghĩa là số lượng các giá trị thuộc tính như nhau và các giá trị như nhau đối với thuộc tính A_j phải tồn tại trong cả D và D' , làm cho việc trao đổi giữa các bản ghi dễ dàng hơn.

Tùy thuộc vào cách thức đưa ra các thông kê mà kỹ thuật chuyển đổi dữ liệu có thể được áp dụng một cách tự nhiên hay không. Với các SDB chuyên dụng dưới dạng bảng (thông kê vĩ mô), thì SDB gốc được biến đổi một cách tự nhiên thành một SDB biến đổi d-chiều, bằng cách chuyển đổi các giá trị thuộc tính giữa các bản ghi. Tuy nhiên, điều này không phù hợp với các cơ sở dữ liệu có mục đích chung, ở đó tính đúng đắn và tính chính xác của các thông tin được đưa ra là yêu cầu cơ bản, đặc biệt là với các truy vấn không phải truy vấn thống kê.

Một SDB D là một SDB có thể biến đổi *d-chiều* nếu các điều kiện sau thỏa mãn :

- Các bản ghi của D phải có ít nhất $m \geq d+1$ thuộc tính.
- D phải chứa ít nhất $N \geq (\bar{M}/2)2^d$ bản ghi, với \bar{M} là kích cỡ tối đa của các tập $M_j, j = 1, 2, \dots, M$ các giá trị có thể của A_1, A_2, \dots, A_M của D .

Một kỹ thuật mở rộng cho Chuyển đổi dữ liệu đã được nghiên cứu, được gọi là *chuyển đổi dữ liệu xấp xỉ*, định nghĩa cho các thuộc tính logic, áp dụng cho các SDB đưa ra các thông kê vĩ mô. Một SDB biến đổi D' của SDB gốc D được sinh ra ngẫu nhiên, xét đến một phần của D , với các giá trị được lấy từ một phân phối của các thông kê bậc k của SDB gốc D .

Tóm lại, mức độ bảo vệ của kỹ thuật chuyển đổi dữ liệu sẽ tăng nếu giảm d , đồng nghĩa với việc giảm các thông tin thống kê mà người dùng yêu cầu. Nói chung kỹ thuật này vẫn chưa được phát triển đầy đủ cho các SDB tĩnh và động.

5.5.3.1.1 Các câu truy vấn mẫu ngẫu nhiên (Random-sample queries)

Cục điều tra dân số Mỹ sử dụng kỹ thuật *mẫu ngẫu nhiên* để ngăn chặn suy diễn trong các cơ sở dữ liệu thống kê. Ý tưởng của kỹ thuật này là sử dụng các mẫu bản ghi từ các SDB động, các mẫu này được lấy từ các tập truy vấn tương ứng với các truy vấn thống kê, thay vì lấy mẫu trong toàn bộ SDB.

Cơ chế cơ bản của kỹ thuật này là thay thế tập truy vấn (có liên quan đến một câu truy vấn thống kê) bằng một tập truy vấn được lấy mẫu (*sampled query set*) gồm một tập con các bản ghi được chọn lựa chính xác trong tập truy vấn gốc. Sau đó, tiến hành tính toán thống kê yêu cầu trên tập truy vấn mẫu này. Sử dụng một hàm chọn $f(C, i)$ để chọn lựa các bản ghi từ tập truy vấn gốc tương ứng với thống kê $q(C)$ mà người dùng yêu cầu.

Giả sử, C là một công thức đặc trưng, $X(C)$ là tập truy vấn tương ứng với C và $q(C)$ là một truy vấn thống kê mà người dùng yêu cầu.

Khi đó, hàm f được xác định như sau: mỗi bản ghi i nằm trong tập truy vấn $X(C)$ có một xác suất được chọn lấy mẫu là p , p có thể do người quản trị cơ sở dữ liệu định nghĩa. Hàm chọn f được thực hiện, bằng cách sử dụng 2 hàm $r(i)$ và $g(C)$, trong đó hàm $r(i)$ ánh xạ bản ghi thứ i vào một dãy ngẫu nhiên $m \geq k$ bit và hàm $g(C)$ ánh xạ đặc trưng C vào một xâu có độ dài m , trên bảng chữ cái $\{0,1,*\}$, trong đó '*' chỉ ra "sự không quan tâm". Xâu này bao gồm chính xác k bit và $m-k$ dấu sao. Để thực hiện các hàm r và g nên sử dụng các thuật toán mã hoá (ví dụ DES).

Bản ghi x_i (trong tập truy vấn được lấy mẫu) được xác định trên cơ sở ánh xạ giữa $r(i)$ và $g(C)$. Một tương ứng tồn tại giữa $r(i)$ và $g(C)$ khi mỗi ký hiệu (khác với '*') trong $g(C)$ giống hệt ký hiệu tương ứng trong $r(i)$.

Việc định nghĩa hàm $f(C, i)$ phụ thuộc vào giá trị của xác suất p .

- Nếu $p < 1/2$, hàm được định nghĩa như sau:

$$f(C, i) = \begin{cases} 1 & \text{nếu } r(i) \text{ tuong ung voi } g(C) \\ 0 & \text{nếu nguoc lai} \end{cases}$$

- Nếu $p > 1/2$

$$f(C, i) = \begin{cases} 0 & \text{nếu } r(i) \text{ tuong ung voi } g(C) \\ 1 & \text{nếu nguoc lai} \end{cases}$$

Tập truy vấn được lấy mẫu $X^*(C)$ bao gồm tất cả các bản ghi được chọn lựa từ $X(C)$, bằng cách sử dụng hàm $f(C, i)$, như sau:

$$X^*(C) = \{ i \in X(C) \mid f(C, i) = 1 \}$$

Xác suất lấy mẫu p có thể không phụ thuộc vào kích cỡ của tập truy vấn (xác xuất không đổi) khi đó mọi câu truy vấn thống kê của người dùng đều lấy xác suất p để chọn lựa tập truy vấn mẫu, hoặc p có thể phụ thuộc vào tập truy vấn.

Nếu p không đổi và rất lớn, chúng ta nên đưa ra giới hạn kích cỡ tập truy vấn tối thiểu, nhằm đối phó với các tập truy vấn có kích cỡ nhỏ. Đúng như vậy, với các tập truy vấn nhỏ cùng với xác suất p lớn dẫn đến có thể lựa chọn tất cả các bản ghi trong tập truy vấn này, tạo điều kiện cho khả năng suy diễn. Mặt khác, khi các giá trị p nhỏ cũng không phù hợp vì nó làm cho tập truy vấn mẫu có kích cỡ rất nhỏ, không đảm bảo được tính chính xác khi đưa ra các thống kê, do đó cần phải đưa vào tập truy vấn mẫu một số lượng lớn các bản ghi.

Sau đây là một số cách để chọn lựa xác suất lấy mẫu p .

Giải pháp 1: lựa chọn một xác suất biến đổi p , giảm theo kích cỡ của tập truy vấn. Để thực hiện giải pháp này, ta cần dự tính trước các xác suất khác nhau, sau đó chọn lựa xác suất nào phụ thuộc vào cách tổ chức các bản ghi trong SDB. Như vậy, với mỗi tập truy vấn ta sẽ chọn xác suất p tương ứng với kích cỡ của nó, sau đó tính toán thống kê yêu cầu trên tập truy vấn được lấy mẫu.

Giải pháp 2: tính toán một số câu trả lời cho thống kê yêu cầu, tương ứng với các giá trị khác nhau của p và chọn ra một câu trả lời trong số các câu trả

lời đã được tính toán, dựa trên cơ sở kích cỡ tập truy vấn của thống kê yêu cầu.

Giải pháp 3: gán cho p một giá trị, giá trị này tỷ lệ nghịch với số lượng các bản ghi được xem xét (đây là các bản ghi đứng trước bản ghi đầu tiên trong tập truy vấn được lấy mẫu, bản ghi đầu tiên này do DBA chọn).

Nhược điểm của kỹ thuật lấy mẫu nhiên là những câu truy vấn tương đương về mặt logic có thể đưa ra kết quả là các tập truy vấn khác nhau.

Để đảm bảo tính nhất quán của các thống kê được đưa ra, chúng ta nên định nghĩa *hàm chọn* sao cho các thống kê tương đương trả lại cùng một kết quả, nhằm ngăn chặn các tấn công tính trung bình. Bằng cách lặp lại câu truy vấn tương tự nhiều lần và tính trung bình kết quả, kẻ tấn công có thể ước lượng được bias-độ lệch được đưa vào trong việc lấy mẫu, sau đó áp dụng các kỹ thuật suy diễn chuẩn. Ngoài ra, để đảm bảo cho các câu truy vấn khác nhau nhưng tương đương về mặt ngữ nghĩa phải sinh ra các tập truy vấn giống nhau, ta cũng nên đưa các công thức đặc trưng về dạng thông thường, bằng cách giới hạn cú pháp của các câu truy vấn (Ví dụ đưa các công thức đặc trưng về dạng tổ hợp của phép toán logic AND).

Tiếp theo, chúng ta xem xét tính đúng đắn của kỹ thuật lấy mẫu nhiên các câu truy vấn. Trước hết với *hàm chọn* $f(C, i)$, các tần số tương đối và giá trị trung bình được tính toán trên các *tập truy vấn mẫu* như sau:

$$Rfreg^*(C) = (|X^*(C)|) / (pN)$$

$$Avg^*(C, A_j) = \frac{1}{|X^*(C)|} \sum_{i \in X^*(C)} x_{ij}$$

Giá trị mong muốn của $|X^*(C)|$ là $p |X(C)|$ và giá trị mong muốn của tần số lấy mẫu là $|X(C)| / N$, hay đây chính là tần số thực. Các giá trị của p và N có thể do người quản trị cơ sở dữ liệu thông báo, để cho phép người sử dụng đánh giá sự chính xác của các ước lượng thu được. Bằng cách sử dụng các giá trị biết trước này, người sử dụng cũng có thể tính toán ra được các thống kê *Count*, *Sum* được lấy mẫu, và cả các thống kê thực *Count* và *Sum*, bằng cách

khai thác các giá trị $Rfreq^*$ và Avg^* và các quan hệ tồn tại giữa các thống kê này.

Kỹ thuật lấy mẫu ngẫu nhiên các câu truy vấn có một lỗi trong khi tính toán thống kê, nguyên nhân là do các thống kê được tính toán trên các tập truy vấn được lấy mẫu.

Lỗi tương đối (*relative error*) f_e giữa tần số lấy mẫu và tần số thực được xác định như sau:

$$f_e = \frac{Rfreq^*(C) - Rfreq(C)}{Rfreq(C)} \text{ trong đó, } Rfreq(C) = |X(C)| / N$$

$$\Rightarrow f_e = 0$$

Tần số tương đối mẫu là một ước lượng không chêch của tần số tương đối thực, bởi vì lỗi tương đối f_e bằng 0.

Nói chung, với các tập truy vấn có kích thước nhỏ, lỗi khó có thể được chấp nhận. Có thể giảm lỗi bằng cách tăng giá trị của p , tuy nhiên, các giá trị của p cao làm tăng rủi ro bị lộ, vì khi đó xác suất của một bản ghi (trong một tập truy vấn được lấy mẫu) sẽ tăng. Sau đó, ta nên đưa ra giới hạn kích cỡ tập truy vấn tối thiểu để tránh các lỗi khó chấp nhận được trong các thống kê được đưa ra, làm ảnh hưởng đến sự chính xác của chúng.

Lỗi tương đối $a_{e,j}$ giữa trung bình lấy mẫu và trung bình thực được xác định như sau:

$$A_{e,j} = \frac{Avg^*(C, A_i) - Avg(C, A_i)}{Avg(C, A_i)}$$

Sau đây, ta sẽ xét một số kiểu tần công đối với kỹ thuật lấy mẫu ngẫu nhiên.

- *Kiểu tần công 1:*

Kết tần công yêu cầu một tập các truy vấn thống kê tương đương với truy vấn thống kê $q(C)$, nghĩa là cùng có tập truy vấn là $X(C)$, nhưng các truy vấn này lại có các tập truy vấn mẫu khác nhau. Sau đó lấy trung bình các kết quả

của các câu truy vấn đó mà hệ thống trả lại, để suy diễn ra kết quả của truy vấn $q(C)$ cần tìm.

Cụ thể, ta giả sử $q(C_1), \dots, q(C_m)$ là các truy vấn thống kê tương đương với $q(C)$, nghĩa là các đặc trưng C_1, \dots, C_m cùng xác định một tập truy vấn $X(C)$, nhưng các tập truy vấn mẫu của chúng khác nhau:

$$C_1, \dots, C_m / X(C_i) = X(C), \text{ và } X^*(C_i) \neq X^*(C), \quad i=1, \dots, m$$

Khi đó, có thể thu được một ước lượng $\hat{q}(C)$ của $q(C)$:

$$\hat{q}(C) = \frac{1}{m} \sum_{i=1}^m q^*(C_i)$$

Như vậy, bằng cách tính trung bình các giá trị kết quả được lấy mẫu từ các thống kê tương đương $q(C_1), \dots, q(C_m)$ ta thu được $q(C)$.

Ví dụ, thông kê $q(C) = q((Sex=M) \wedge (Dept_code = Dept1))$, ta có các thống kê tương đương sau:

$$q(C_1) = q(Sex = \neg F) \wedge (Dept_code = Dept1)$$

$$q(C_2) = q(Sex = M) \wedge \neg(Dept_code = Dept2) \vee (Dept_code = Dept3))$$

Để ngăn chặn kiểu tấn công này, hàm g cần biến đổi công thức đặc trưng về dạng thông thường, đảm bảo rằng các công thức đặc trưng tương đương sinh ra một tập truy vấn mẫu giống nhau ($g(C)=g(D)$ nếu C và D tương đương). Nhưng vẫn đề biến đổi một công thức đặc trưng về dạng thông thường cũng khá khó khăn.

- *Kiểu tấn công 2:*

Tính trung bình các kết quả thu được từ các thông kê tương ứng với các tập con rời nhau của $X(C)$.

Giả sử $q(C_1), \dots, q(C_m)$ là các thống kê, trong đó C_1, \dots, C_m là các tập con rời nhau của tập truy vấn $X(C)$:

$$X(C_i) \cap X(C_j) = \emptyset, \quad i \neq j$$

$$\text{Và } X(C) = X(C_1) \cup \dots \cup X(C_m)$$

Khi đó, thu được một ước lượng của $q(C)$ như sau:

$$\hat{q}(C) = \frac{1}{m} \sum_{i=1}^m \hat{q}(C_i)$$

Trong đó, $\hat{q}(C_i)$ là các ước lượng của các thông kê $q(C_i)$.

Người ta đã chứng minh được rằng, với kiểu tấn công thứ nhất, chỉ cần cận dưới m - số lượng các truy vấn cần thiết để thu được một ước lượng chính xác của giá trị thực của một thông kê tần suất tương đối $Rfreq$ là:

$$m \geq (3.92)^2 \left(\frac{1-p}{p}\right) |X(C)| > 15 |X(C)| \left(\frac{1-p}{p}\right)$$

Nếu xác suất lấy mẫu p là cố định, thì hàm trên sẽ tăng tuyến tính với kích cỡ $|X(C)|$. Vì vậy, với các tập truy vấn nhỏ, chỉ cần một vài câu truy vấn có thể suy ra được ước lượng của thông kê cần tìm, còn với những tập truy vấn lớn sẽ đòi hỏi số lượng truy vấn lớn hơn. Ví dụ, với $p = 0.5$, chỉ cần 10 câu truy vấn với một tập truy vấn kích cỡ $|X(C)| = 10$, và cần 450 câu truy vấn với một tập truy vấn kích cỡ 100. Do đó, cần phải có kỹ thuật giới hạn kích cỡ tập truy vấn để giải quyết các vấn đề liên quan đến các tập truy vấn có kích cỡ nhỏ.

Tóm lại với các tập truy vấn lớn thì kỹ thuật lấy mẫu ngẫu nhiên có thể bảo vệ chống lại các tấn công kiểu theo dõi, đồng thời có thể chống lại các tấn công lấy trung bình các kết quả được lấy mẫu, vì số lượng các truy vấn yêu cầu để đạt được một ước lượng chính xác của truy vấn cần tìm là rất lớn. Ngoài ra, sử dụng một hàm lấy mẫu g biến đổi các công thức đặc trưng về dạng thông thường, cũng giúp ta nhận biết các công thức đặc trưng tương đương, để tránh kiểu tấn công lấy trung bình kết quả của các thông kê tương đương. Tuy nhiên, kiểu tấn công lấy trung bình các kết quả thông kê của các tập truy vấn con rời nhau có thể vẫn là một vấn đề cần quan tâm, do đó cần thiết sử dụng cơ chế kiểm toán để lưu các theo dõi về các tập truy vấn đã được yêu cầu.

5.5.3.1.2 Kỹ thuật gây nhiễu cố định (fixed perturbation)

Giống như kỹ thuật chuyên đổi dữ liệu, kỹ thuật nhiễu cố định tạo ra một cơ sở dữ liệu được sửa đổi so với cơ sở dữ liệu ban đầu. Cơ sở dữ liệu sửa đổi này sẽ được tạo ra bởi việc thay đổi giá trị của mỗi trường bằng một giá trị nhiễu được sinh ra ngẫu nhiên.

Bởi vì nhiễu được thực hiện chỉ một lần, nên các truy vấn lặp đi lặp lại sẽ có kết quả là các giá trị tương thích. Do đó, kiểu tấn công lấy trung bình các kết quả sẽ không thể thu được giá trị ban đầu của dữ liệu.

Kỹ thuật này bao gồm: sửa đổi giá trị của các thuộc tính và tính toán các thống kê dựa vào các giá trị gây nhiễu này (Traub et al..1984). Với mỗi câu truy vấn, việc gây nhiễu không khác nhau.

Kỹ thuật này được nghiên cứu cho các thống kê tuyến tính và cho các thuộc tính số.

Thống kê tuyến tính là một thống kê $q(C)$ có dạng:

$$q(C) = \sum_{i \in X(C)} x_{ij}$$

Trong đó x_{1j}, \dots, x_{mj} là các giá trị của một thuộc tính số A_j dành cho mỗi bản ghi trong số m bản ghi của tập truy vấn $X(C)$.

Cho N là kích cỡ của SDB và ta xét thuộc tính A_j . Tuỳ thuộc vào lược đồ gây nhiễu, mỗi giá trị thực x_{ij} (với $i = 1, \dots, N$) của một thuộc tính A_j bị thay thế bằng một giá trị gây nhiễu x'_{ij} .

$$x'_{ij} = x_{ij} + e_i \text{ với } i = 1, \dots, N$$

Vector $e = (x' - x) = (e_1, \dots, e_N)$ là một vector gây nhiễu ngẫu nhiên với $x = (x_{1j}, \dots, x_{Nj})$, $x' = (x'_{1j}, \dots, x'_{Nj})$ là các vector của giá trị thực và giá trị gây nhiễu của các bản ghi trong SDB, dành cho thuộc tính A_j . Mỗi thành phần của vector $e(e_i)$ là các biến ngẫu nhiên độc lập tuyến tính, và giá trị mong đợi của vector e là vector 0. Mỗi thành phần của vector e có giá trị trung bình và phương sai mong muốn như sau:

$$E(e_i) = 0, \text{Var}(e_i) = \sigma^2$$

Như vậy, các giá trị nhiễu thường được phân phối với $\mu=0$, và độ lệch chuẩn là σ^2 . Chúng ta hãy giả thiết rằng, các giá trị gây nhiễu e_i được phân bố đều và chúng ta xét một thông kê tuyênlính $q(C)$. Khi đó, xác suất lỗi trong một câu truy vấn vượt quá giá trị giới hạn ε cho trước là:

$$P(|q'(C) - q(C)| \geq |\varepsilon| |X(C)|) \leq \frac{\sigma^2}{(|X(C)| \varepsilon^2)}$$

Trong đó, $q'(C)$ chỉ ra thông kê được tính toán bằng cách sử dụng các giá trị gây nhiễu.

Xác suất phụ thuộc vào phương sai gây nhiễu σ^2 , giới hạn lỗi ε và kích thước $|X(C)|$ của tập truy vấn của thông kê $q(C)$.

Chúng ta quan sát và thấy rằng xác suất lỗi giảm nếu kích cỡ của tập truy vấn tăng. Do vậy, với các tập truy vấn có kích cỡ lớn hơn thì độ chính xác của thông kê tương ứng cao hơn.

Tuy nhiên, trong các trường hợp cho trước một vector gây nhiễu x' (vector gây nhiễu cố định), thì độ chênh lệch $|q'(C) - q(C)|$ có thể khá lớn, dù một số tập truy vấn có kích cỡ rất lớn. Như vậy, trong trường hợp này, với vector gây nhiễu cố định x' , thì dù $|X(C)|$ có kích cỡ rất lớn nhưng độ lệch giữa $q(C)$ và $q'(C)$ vẫn cao, cho nên cần có biện pháp để tăng độ chính xác của thông kê gây nhiễu đưa ra. (Chú ý, trong trường hợp này $q'(C)$ là cố định do x' cố định).

Để đối phó với các trường hợp này, một giải pháp được đưa ra là kiểm tra tổng số lỗi $|q'(C) - q(C)|$ để biết được khi nào nó vượt quá giới hạn cho trước ($|\varepsilon| |X(C)|$). Nếu lỗi $|q'(C) - q(C)|$ quá lớn và đồng thời $|X(C)|$ đã lớn hơn một giá trị giới hạn xác định nào đó, n_0 (giá trị này do DBA định nghĩa), thì sử dụng *cơ chế hiệu chỉnh*. Cơ chế này như sau: bổ sung thêm vào thông kê thực $q(C)$ một biến số ngẫu nhiên $Per(C)$ - là một vector, nó có giá trị trung bình bằng 0, và phương sai σ_1^2 , và được chọn lựa nên $|Per(C)|$ thỏa mãn giới hạn lỗi.

Ta giả sử $Per(C) = (per1, per2, \dots, perN)$, khi đó ta vẫn có vector gây nhiễu cố định $x' = (x'_{1j}, \dots, x'_{Nj})$ và $x'_{ij} = x_{ij} + e_i$ với $i = 1, \dots, N$.

Tuy nhiên, lúc này $x_{ij} = x_{ij} + per_i$, $i = 1, \dots, N$. (Xét với thuộc tính A_j), đồng thời per_i có thể âm hoặc dương.

Khi đó, độ lệch giữa thống kê thực và thống kê gây nhiễu sẽ thỏa mãn điều kiện: $|q'(C) - q(C)| \leq |\varepsilon| |X(C)|$

Việc gây nhiễu có thể được áp dụng cố định với các giá trị thuộc tính được sửa đổi thường xuyên trong SDB, hoặc áp dụng động, có nghĩa là các giá trị gây nhiễu được tính toán tại thời điểm được sử dụng cho việc tính toán câu truy vấn. Chế độ thứ 2 phù hợp cho các SDB mục đích chung hơn, ở đó các ứng dụng thông thường sử dụng SDB với nhiều mục đích.

Các kỹ thuật gây nhiễu cố định cũng được nghiên cứu cho các thuộc tính không số (Warner, 1965; Abul-Ela, 1967). Với các kỹ thuật này, DBA cần đảm bảo sự cân bằng giữa độ chính xác của các thống kê được đưa ra và mức độ bảo vệ cho SDB, bằng cách thiết lập chính xác các tham số để thực hiện gây nhiễu cho các giá trị thuộc tính.

5.5.3.1.3 Gây nhiễu dựa vào truy vấn

Nhiễu dựa vào truy vấn là một kỹ thuật gây nhiễu dữ liệu không yêu cầu tạo một cơ sở dữ liệu ủy nhiệm. Với mỗi truy vấn được tạo ra trong SDB, một hàm gây nhiễu sẽ được áp dụng với tất cả các thuộc tính ảnh hưởng đến giá trị kết quả. Giá trị bias được đưa vào hàm đó sẽ không được lưu trong cơ sở dữ liệu, và giá trị này cũng khác nhau so với các truy vấn riêng.

Giả sử x_{ij} là một giá trị của thuộc tính A_j tương ứng với bản ghi thứ i trong SDB. Với một thống kê $q(C)$, kỹ thuật này sẽ áp dụng một hàm sửa đổi f cho mỗi giá trị thuộc tính x_{ij} của các bản ghi thuộc vào tập truy vấn $X(C)$ của $q(C)$ để sinh ra một giá trị sửa đổi $x'_{ij} = f(x_{ij})$, sử dụng để tính toán $q(C)$.

Trước khi mô tả chi tiết, chúng ta đưa ra định nghĩa “*làm lô - compromise*” được dùng trong kỹ thuật này. Khi áp dụng kỹ thuật sửa đổi này, người dùng chỉ biết một ước lượng \hat{x}_{ij} của giá trị thực x_{ij} của một thuộc tính. Giá sử $\sigma(\hat{x}_{ij})$ là độ lệch chuẩn của ước lượng đó, và giá trị trung bình mong đợi là $E(\hat{x}_{ij}) = x_{ij}$.

- Khi đó, SDB được gọi là *có thể bị lộ* nếu một người dùng đạt được một ước lượng \hat{x}_{ij} của giá trị thực x_{ij} của một thuộc tính A_j , sao cho:

$$\sigma(\hat{x}_{ij}) < f(x_{ij})$$

Trong đó, f là một hàm sửa đổi phụ thuộc vào giá trị x_{ij} .

- Một SDB được gọi là *có thể bị lộ thông kê* nếu:

$$\sigma(\hat{x}_{ij}) < c / |x_{ij} - \bar{x}_j|$$

Trong đó, c là một hằng số của SDB do DBA lựa chọn, và \bar{x}_j là giá trị trung bình của thuộc tính A_j .

Chúng ta sẽ mô tả hàm sửa đổi và chỉ ra kỹ thuật này sẽ thực hiện như thế nào đối với các thống kê *Sum* và *Count*.

- *Thông kê Sum*:

Xét thống kê *Sum* sau: $q(C) = \text{Sum}(C, A_j)$, giả sử n là số lượng các bản ghi của tập truy vấn $X(C)$. Giá trị trung bình của A_j trong tập $X(C)$ ký hiệu là \bar{x}_{C_j}

Giá trị trung bình của A_j trên toàn bộ SDB được ký hiệu là \bar{x}_j . Thông kê tổng $\text{Sum}(C, A_j)$ được tính trên các giá trị sửa đổi x'_{ij} của thuộc tính A_j trong n bản ghi của $X(C)$, tạo ra giá trị tổng được làm nhiễu sau:

$$S' = \sum_{i=1}^n x'_{ij}, \text{ với } x'_{ij} = f(x_{ij}) = x_{ij} + z_1 (x_{ij} - \bar{x}_{C_j}) + z_2$$

Trong đó, z_1 và z_2 là các biến ngẫu nhiên độc lập được sinh ra cho mỗi bản ghi. Các giá trị trung bình và phương sai của z_1 và z_2 được mong đợi như sau:

$$\begin{aligned} E(z_1) &= 0, \quad \text{Var}(z_1) = 2a^2 \\ E(z_2) &= 0, \quad \text{Var}(z_2) = \frac{2a^2}{n} (\bar{x}_{C_j} - \bar{x}_j)^2 \end{aligned}$$

Trong đó, a là một hằng số đối với tất cả các câu truy vấn. Kết quả được sửa đổi S' là một ước lượng không chênh của giá trị thực S , vì:

$$E(S') = E\left(\sum_{i=1}^n x'_{ij}\right) = \sum_{i=1}^n x_{ij} = S$$

Điều này cho thấy là các giá trị trung bình của các biến ngẫu nhiên z_1 và z_2 bằng 0. Phương sai của S' như sau:

$$\text{Var}(S') = 2na^2 s_c^2 + 2a^2 (\bar{x}_{c_j} - \bar{x}_j)^2$$

Trong đó, $s_c^2 = \sum (x_{ij} - \bar{x}_{c_j})^2 / n$ là phương sai mẫu được tính trên tập $X(C)$.

Phương sai của S' có đặc điểm là tăng lên khi các giá trị x_{ij} trong tập truy vấn $X(C)$ khác nhiều với các giá trị khác (vì khi đó s_c^2 sẽ tăng), và phương sai này cũng tăng khi các giá trị x_{ij} trong $X(C)$ khác xa so với giá trị trung bình \bar{x}_j (làm cho $(\bar{x}_{c_j} - \bar{x}_j)^2$ tăng).

Cận dưới cho độ lệch tiêu chuẩn của ước lượng S' như sau:

$$\sigma(S') > a |x_{ij} - \bar{x}_j|$$

với mỗi bản ghi i trong tập truy vấn $X(C)$.

- *Thống kê Count:*

Đối với thống kê *Count*, giả sử n là kích cỡ tập truy vấn. Kỹ thuật này phải ngăn chặn người dùng đạt được ước lượng của n là \hat{n} , nghĩa là \hat{n} phải thỏa mãn:

$$\sigma_{\hat{n}} < c_1$$

Trong đó c_1 là một giá trị lớn vừa đủ sao cho từ kết quả làm nhiễu \hat{n} không suy diễn ra được n .

Giá trị được làm nhiễu \hat{n} do thống kê *Count* trả về được tính như sau:

$$\hat{n}' = \sum_{j=3}^n z_3$$

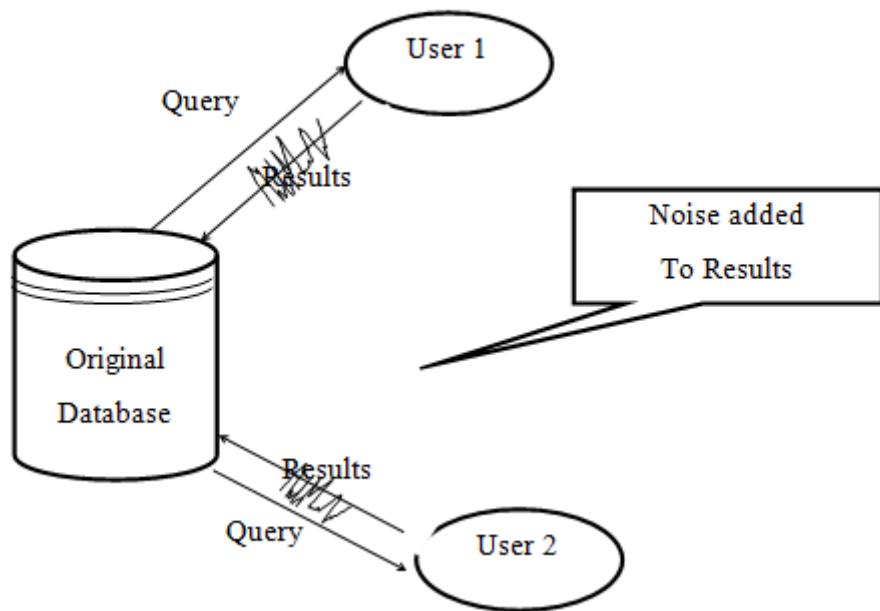
với $E(z_3) = 1$ và $Var(z_3) = a_1^2/n$, với giả thiết rằng các giá trị của z_3 được sinh độc lập với các bản ghi x_i trong tập truy vấn $X(C)$ và với mỗi câu truy vấn. Các giá trị của z_3 được lưu trong SDB.

Ta có: $E(n') = n$ và $Var(n') = a_1^2$

Theo kỹ thuật này, thì số lượng các truy vấn cần thiết để làm lộ SDB phải ít nhất bằng a_1/c_1 câu truy vấn.

5.5.3.2 Kỹ thuật gây nhiễu đầu ra

Các kỹ thuật gây nhiễu đầu ra thực hiện sửa đổi trên các kết quả được tính toán chính xác của một câu truy vấn thống kê yêu cầu, trước khi chuyển nó cho người sử dụng. Về cơ bản, kỹ thuật này dựa trên một số kiểu làm tròn trên các kết quả chính xác.



Hình 5.19. Mô tả kỹ thuật gây nhiễu đầu ra

Làm tròn (rounding)

Kỹ thuật này tiến hành gây nhiễu kết quả được tính cho một thống kê, và nó được xem là một kỹ thuật gây nhiễu đầu ra.

Khái niệm "*làm tròn*" xuất phát từ trong thực tế, các giá trị kết quả được làm tròn trước khi được đưa ra. Chính xác hơn, kết quả thực Q dành cho một thống kê $q(C)$ được làm tròn lên hoặc làm tròn xuống thành bội số gần nhất của một cơ số b cụ thể nào đó, thông qua một hàm làm tròn, $r(Q)$, sinh ra một kết quả sửa đổi, $Q' = r(Q)$. Tuỳ thuộc vào hàm làm tròn mà ta có các kiểu làm tròn được định nghĩa như sau:

- *Làm tròn có hệ thống* (*systematic rounding*)
- *Làm tròn ngẫu nhiên* (*random rounding*)
- *Làm tròn có kiểm soát* (*controlled rounding*)

Giả thiết:

Q' là một kết quả sửa đổi, nó được tính toán cho thống kê yêu cầu $q(C)$.

$b' = \lfloor (b+1)/2 \rfloor$ (ký hiệu $\lfloor \rfloor$ chỉ làm tròn xuống số nguyên gần nhất)

$$d = Q \bmod b.$$

Bây giờ chúng ta mô tả hàm làm tròn được định nghĩa như thế nào cho từng kiểu làm tròn.

- *Làm tròn có hệ thống*, $r(q)$ được định nghĩa như sau:

$$r(Q) = \begin{cases} Q & \text{nếu } d = 0 \\ Q - d & \text{nếu } d < b \\ Q + b - d & \text{nếu } d \geq b \end{cases}$$

Từ định nghĩa của $r(Q)$, người sử dụng có thể suy diễn được giá trị thực Q nằm trong khoảng:

$$[Q' - b' + 1, Q' + b' - 1]$$

Trong các trường hợp đặc biệt, *làm tròn có hệ thống* cho phép người sử dụng suy ngược trở lại giá trị thực của các kết quả. Ví dụ, với một tập các truy vấn *Sum* được yêu cầu trên các tập truy vấn rời nhau, và mối quan hệ giữa các khoảng kết quả của những câu truy vấn đó.

- *Làm tròn ngẫu nhiên*, $r(Q)$ được định nghĩa như sau:

$$r(Q) = \begin{cases} Q \text{ nếu } d = 0 \\ Q - d \text{ với xác suất } 1 - p \\ Q + b - d \text{ với xác suất } p \end{cases}$$

Nếu $p = d/b$, thì kiểu làm tròn này không chêch. Tuy nhiên, SDB vẫn có thể bị lộ, do các tấn công dựa vào việc tính trung bình các kết quả sửa đổi (cho cùng một câu truy vấn), tiến hành lặp đi lặp lại nhiều lần, cũng như các tấn công dựa vào trình theo dõi.

Nói chung, với làm tròn có hệ thống và làm tròn ngẫu nhiên, sau khi đã làm tròn từng thống kê trên các tập truy vấn tách rời, rồi mới tính tổng thì kết quả thu được có thể khác với kết quả thu được sau làm tròn các thống kê trên hợp của các tập truy vấn đó. Do vậy, chúng ta cần nghiên cứu kỹ thuật làm tròn có kiểm soát.

Làm tròn có kiểm soát được áp dụng cho các SDB có dạng bảng. Với kỹ thuật làm tròn này, bất cứ khi nào ta *làm tròn có hệ thống* hoặc *làm tròn ngẫu nhiên* trên một ô x_{ij} của bảng, thì áp dụng làm tròn tương tự như vậy với 3 ô khác của bảng, sao cho bảo toàn các giá trị thực của các thống kê tương ứng với hàng thứ i và cột thứ j của bảng. Nghĩa là, nếu theo *làm tròn hệ thống* và *làm tròn ngẫu nhiên*, ô x_{ij} được cộng hoặc trừ một lượng d_{ij} , thì theo *làm tròn có kiểm soát*, cũng lượng d_{ij} này sẽ được cộng (hoặc trừ) vào 3 ô khác, sao cho khi tính tổng hàng i và tổng cột j , kết quả đưa ra bằng với tổng thực của hàng i và cột j .

Giả sử, $X(C_1), \dots, X(C_n)$ là các tập truy vấn tách rời, $X(C_{n+1}) = X(C_1) \cup \dots \cup X(C_n)$ là tập truy vấn hợp nhất. $r(q_1), \dots, r(q_n)$ là các kết quả làm tròn của từng thống kê (được tính toán trên từng tập truy vấn) và $r(q_{n+1})$ là thống kê được tính toán trên tập truy vấn $X(C_{n+1})$.

Thông qua việc sử dụng *làm tròn có hệ thống* và *làm tròn ngẫu nhiên*, chúng ta có:

$$r(q_1) + \dots + r(q_n) \neq r(q_{n+1})$$

Thông qua việc sử dụng làm tròn có kiểm soát, chúng ta có:

$$r(q_1) + \dots + r(q_n) = r(q_{n+1})$$

Tóm lại, các kỹ thuật làm tròn cung cấp bảo vệ mức thấp cho SDB và chúng vẫn bị tấn công bởi nhiều kiểu tấn công như: tính trung bình, trình theo dõi. Do đó, chúng cần được sử dụng kết hợp với các kỹ thuật bảo vệ khác.

5.6 SO SÁNH CÁC KỸ THUẬT CHỐNG SUY DIỄN

Trong phần này, trình bày một bộ tiêu chuẩn để so sánh tất cả các kỹ thuật chống suy diễn đã nghiên cứu. Các kỹ thuật chống suy diễn có thể được phân loại và so sánh tuỳ thuộc vào các đặc trưng và tiêu chuẩn liên quan:

- An toàn (*security*): chỉ ra mức bảo vệ do kỹ thuật cung cấp. Mức bảo vệ này được đánh giá, thông qua việc kiểm tra thử nghiệm kỹ thuật bảo vệ chống lại nhiều kiểu tấn công (ví dụ, tấn công dựa vào các trình theo dõi, các hệ tuyến tính) để tìm ra khi nào thì xảy ra tình trạng lộ và lộ chính xác hay lộ từng phần. Việc khai thác SDB sẽ tăng nếu người sử dụng có các kiến thức phụ liên quan đến lược đồ và các nội dung của SDB. Tiêu chuẩn sau đây được sử dụng khi phân loại các kỹ thuật bảo vệ, liên quan đến tính an toàn:
 - *Lộ chính xác (Exact compromise)*: chỉ ra khả năng làm lộ chính xác SDB của một người sử dụng, khi kỹ thuật được sử dụng.
 - *Lộ từng phần (Partial compromise)*: chỉ ra khả năng làm lộ từng phần SDB của một người sử dụng, khi kỹ thuật được sử dụng.
 - *Robustness*: Chỉ ra khả năng của kỹ thuật (trong việc chống suy diễn) quan tâm đến các kiến thức phụ mà người sử dụng có thể có.
- *Chất lượng (Quality)*: Chỉ ra chất lượng chung của thông tin được chuyển cho người sử dụng, có sự can thiệp của một kỹ thuật chống suy diễn. Tiêu chuẩn sau được sử dụng khi phân loại các kỹ thuật bảo vệ, liên quan đến chất lượng:
 - *Mất mát thông tin (Information loss)*: Nói chung, nó chỉ ra sự đầy đủ của thông tin được chuyển cho người sử dụng, có sự can thiệp của kỹ thuật. Với các kỹ thuật dựa vào giới hạn, việc mất mát thông tin cần được hạn chế tối đa để đảm bảo chống suy

diễn. Với các kỹ thuật dựa vào gây nhiễu, nó liên quan đến phương sai của lỗi trong các thống kê gây nhiễu được cung cấp cho người sử dụng.

- *Độ lệch (bias)*: Nói chung, áp dụng cho các kỹ thuật dựa vào gây nhiễu. Nó xảy ra khi ước lượng của thống kê thực (được cung cấp cho người sử dụng) có giá trị mong muốn khác với giá trị thực. Người ta thường mong muốn có được ước lượng không chênh của thống kê thực.
- *Độ chính xác (Precision)*: Chỉ áp dụng cho các kỹ thuật dựa vào gây nhiễu. Có thể đánh giá được nó thông qua phương sai của ước lượng được cung cấp cho người sử dụng. Phương sai càng thấp có nghĩa là việc đưa ra các thống kê càng chính xác hơn. Nói cách khác, với phương sai cao hơn, chúng ta cần mức độ bảo vệ của kỹ thuật phải cao hơn. Phải cân đối giữa các yêu cầu này.
- *Tính nhất quán (Consistency)*: Chỉ áp dụng cho các kỹ thuật dựa vào gây nhiễu. Nó chỉ ra rằng việc gây nhiễu không gây ra các mẫu thuẫn và nghịch lý. Ví dụ, mẫu thuẫn xảy ra khi các bản sao của cùng một thống kê, hoặc các thống kê tương đương lại đưa ra các kết quả gây nhiễu khác nhau. Nghịch lý xảy ra khi người sử dụng được trả lại một kết quả không nhất quán với ngữ nghĩa của thống kê yêu cầu.
- *Chi phí (Cost)*: Chỉ ra các chi phí cho việc thực hiện và sử dụng kỹ thuật. Tiêu chuẩn sau đây được sử dụng khi phân loại các kỹ thuật bảo vệ, liên quan đến chi phí:
 - *Chi phí thực hiện (Implementation cost)*: Chỉ ra chi phí cho thực hiện kỹ thuật và thiết lập các tham số yêu cầu.
 - *Tổng chi phí xử lý cho từng câu truy vấn (Processing overhead per query)*: Liên quan đến các yêu cầu về thời gian và lưu giữ của CPU trong quá trình xử lý.

- *Đào tạo người dùng (User education)*: Liên quan đến các chi phí khi đào tạo người sử dụng về kỹ thuật bảo vệ.
- *Sự phù hợp (Suitability)*: Đây chính là sự phù hợp của một bộ tiêu chuẩn với khả năng ứng dụng của một kỹ thuật, liên quan đến kiểu, số lượng các thuộc tính và kiểu của chính SDB. Tiêu chuẩn sau đây được sử dụng để phân loại các kỹ thuật bảo vệ, liên quan đến sự phù hợp:
 - *Các thuộc tính số/không số (Numerical/non-numerical attributes)*: Một kỹ thuật có thể được áp dụng cho một kiểu thuộc tính hoặc cả hai.
 - *Số lượng các thuộc tính (Number of attributes)*: Một kỹ thuật có thể được áp dụng cho một hoặc nhiều thuộc tính của lược đồ SDB.
 - *Các SDB động và trực tuyến (On-line dynamic SDBs)*: Một kỹ thuật có thể thích hợp nhiều hoặc ít trong việc chống suy diễn trong các môi trường động và trực tuyến. Các yêu cầu chính cần được quan tâm trong các môi trường này là: việc tính toán các câu truy vấn thống kê được thực hiện tại thời điểm chúng được yêu cầu và các thống kê phản ánh được các cập nhật mới được thực hiện trên các bản ghi của SDB (chẳng hạn như các phép chèn, xoá, sửa đổi). Do đó, tùy thuộc vào cách làm việc, chúng ta có thể có các kỹ thuật phù hợp thực sự cho SDB có kiểu động và trực tuyến này, các kỹ thuật khác không phù hợp khi chúng được phát triển cho các SDB mục đích đặc biệt.

Việc phân loại các kỹ thuật dựa vào hạn chế và gây nhiễu theo các tiêu chuẩn này được trình bày trong các bảng 4.1, 4.2.

Tiêu chuẩn							
An toàn				Chất lượng			
Kỹ thuật	Lộ chính xác	Lộ tung phàn	Robustness	Mật mã thông tin	Độ lệch	Độ chính xác	Tính tương thích

Kích cỡ tập truy vấn	Có	Có	Thấp	Cao	-	-	-
Kích cỡ tập truy vấn mở rộng	Có	Có	Thấp	Trung bình	-	-	-
Chóng lặp tập truy vấn	Có	Có	Thấp	Rất cao	-	-	-
Dựa vào kiểm toán	Không	Có	Thấp	Trung bình	-	-	-
Tiêu chuẩn S _n /N	Có	Có	Thấp	Trung bình	-	-	-
Phân hoạch	Không	Có	Phụ thuộc vào kích cỡ của các lực lượng nguyên tử	Trung bình Hoặc Cao	Có (với các thực thể giả)	-	-
Giá ô	Không	Có	Thấp	Trung bình	-	-	-
Tiêu chuẩn							
Chi phí							
Sự thích hợp							
Kỹ thuật	Chi phí thực hiện	Tổng chi phí xử lý cho từng câu truy vấn	Đào tạo người dùng	Các thuộc tính số/không số	Số lượng các thuộc tính	Các SDB động và trực tuyến	
Kích cỡ tập truy vấn	Thấp	Thấp	Rất thấp	Cả hai	≥ 1	Trung bình	
Kích cỡ tập truy vấn mở rộng	Cao	Cao	Thấp	Cả hai	≥ 1	Trung bình	
Chóng lặp	Cao	Rất cao	Thấp	Cả hai	≥ 1	Trung bình	

tập truy vấn						
Dựa vào kiểm toán	Cao	Rất cao đối với các SDB lớn	Thấp	Cả hai	1	Thấp
Tiêu chuẩn $S_{n/N}$	Thấp	Thấp	Thấp	Cả hai	≥ 1	Trung bình
Phân hoạch	Trung bình (SDB tĩnh) Rất cao (SDB động)	Rất thấp (SDB tĩnh)	Thấp	Cả hai	≥ 1	Có
Giá ô	Cao	Hoàn toàn không	Hoàn toàn không	Cả hai	≥ 1	Không

Bảng 5.10 So sánh tiêu chuẩn dành cho các kỹ thuật dựa vào hạn chế

Tiêu chuẩn							
An toàn				Chất lượng			
Kỹ thuật	Lộ chính xác	Lộ từng phần	Robustness	Mát mát thông tin	Độ lệch	Độ chính xác	Tính tương thích
Chuyển đổi dữ liệu	Không	Có	Cao	Không	Có	Có thể rất thấp	Cao
Lấy mẫu ngẫu nhiên các câu truy vấn	Có	Có	Trung bình	Thấp	Không	Có thể rất thấp	Thấp
Gây nhiễu cố định	Không	Có	Trung bình	Không	Có	Cân bằng dựa vào an toàn	Trung bình

Gây nhiễu dựa vào câu truy ván	Không	Có	Trung bình	Thấp	Không	Cân bằng dựa vào an toàn	Thấp
Làm tròn có hệ thống	Có	Có	Thấp		Có	Phụ thuộc vào cơ sở làm tròn	Thấp
Làm tròn ngẫu nhiên	Có	Có	Thấp		Không	Phụ thuộc vào cơ sở làm tròn	Trung bình
Làm tròn có kiểm soát		Có	Thấp			Phụ thuộc vào cơ sở làm tròn	Cao

Tiêu chuẩn

Chi phí	Sự thích hợp					
Kỹ thuật	Chi phí thực hiện	Tổng chi phi xử lý cho từng câu truy ván	Đào tạo người dùng	Các thuộc tính số/ không số	Số lượng các thuộc tính	Các SDB động và trực tuyến
Chuyển đổi dữ liệu	Rất cao	Hoàn toàn không	Trung bình	Không số	1	Không
Lấy mẫu ngẫu nhiên các câu truy vấn	Thấp	Thấp	Thấp	Cả hai	1	Cao
Gây nhiễu cộ định	Thấp	Rất thấp	Rất thấp	Số	1	Trung bình
Gây	Trung	Trung	Rất	Số	1 hoặc	Cao

nhiều dựa vào câu truy vấn Làm tròn có hệ thống	bình	bình	cao		>1 nếu không phụ thuộc	
Làm tròn ngẫu nhiên	Thấp	Trung bình	Thấp	Cả hai	1	Thấp/ Trung bình
Làm tròn có kiểm soát	Thấp	Trung bình	Thấp	Cả hai	1	Thấp/ Trung bình
	Trung bình	Trung bình	Thấp	Cả hai	1	Thấp

Bảng 5.11 So sánh tiêu chuẩn dành cho các kỹ thuật dựa vào gây nhiễu

Từ các bảng này, chúng ta có thể rút ra:

Các kỹ thuật dựa vào hạn chế có thể dẫn đến tình trạng mất mát thông tin (ví dụ, kiểm soát chống lặp tập truy vấn) với mục đích đảm bảo chống lộ chính xác và điều này có thể hạn chế quá nhiều tính hữu ích của SDB. Hơn nữa, các kỹ thuật này yêu cầu chi phí khá lớn cho việc xử lý câu truy vấn, đặc biệt với các kỹ thuật dựa vào kiểm toán, phải kiểm tra các câu truy vấn nhằm tránh tình trạng lộ. Nói cách khác, cần có các kiểm soát toàn diện để ngăn chặn một số lượng lớn những người sử dụng khai thác chính xác dãy các câu truy vấn để suy diễn các thông tin bí mật liên quan đến các cá nhân. Người sử dụng thường được cung cấp các thông kê chính xác và nhất quán, chúng được đưa ra khi các thống kê được tính toán trên các giá trị thực của các cá nhân được biểu diễn trong SDB.

Các kỹ thuật dựa vào gây nhiễu cố gắng đưa ra nhiều thông kê hơn, so với các kỹ thuật dựa vào hạn chế, nhằm giảm mất mát thông tin, bằng cách đưa nhiễu vào các thống kê. Tuy nhiên, nhiễu gây ra một vấn đề mới, chẳng hạn như độ lệch và tính tương thích của các thống kê được đưa ra. Tuy nhiên, các

kỹ thuật gây nhiễu dựa vào bản ghi (gây nhiễu dữ liệu) được xem là các kiểm soát suy diễn thích hợp nhất cho các SDB động và trực tuyến. Việc mất mát thông tin liên quan đến phương sai của lỗi. Như đã trình bày ở trên, các chi phí thực hiện và tổng chi phí cho việc xử lý câu truy vấn cũng cần được quan tâm.

Nói chung, chúng ta có thể kết luận rằng không thể tồn tại một kỹ thuật bảo vệ đơn lẻ nào vừa có khả năng cung cấp an toàn cao, vừa ít gây mất mát thông tin và chi phí thấp. Hơn nữa, không có kỹ thuật nào có khả năng ngăn chặn cả tình trạng lộ chính xác và lộ từng phần. Việc chọn lựa một (nhiều) kỹ thuật thích hợp nên được hướng dẫn, thông qua các yêu cầu bảo vệ và các đặc trưng của môi trường cần bảo vệ.

5.7 CÂU HỎI ÔN TẬP

1. Thế nào là cơ sở dữ liệu thống kê? So sánh cơ sở dữ liệu thống kê với cơ sở dữ liệu quan hệ.
2. Viết một vài câu lệnh SQL thực hiện các truy vấn thống kê.
3. Thế nào là lộ chính xác, lộ từng phần?
4. Thế nào là kiến thức làm việc, kiến thức bổ sung.
5. Nếu một số tấn công vào SDB.
6. Mô tả kỹ thuật kiểm soát kích cỡ tập truy vấn.
7. Mô tả kỹ thuật giấu ô
8. Mô tả kỹ thuật gộp.
9. Mô tả kỹ thuật gây nhiễu dữ liệu và gây nhiễu đâu ra.
10. Mô tả tấn công trinh theo dõi và tấn công hệ tuyến tính.
11. So sánh các kỹ thuật kiểm soát suy diễn.

TÀI LIỆU THAM KHẢO

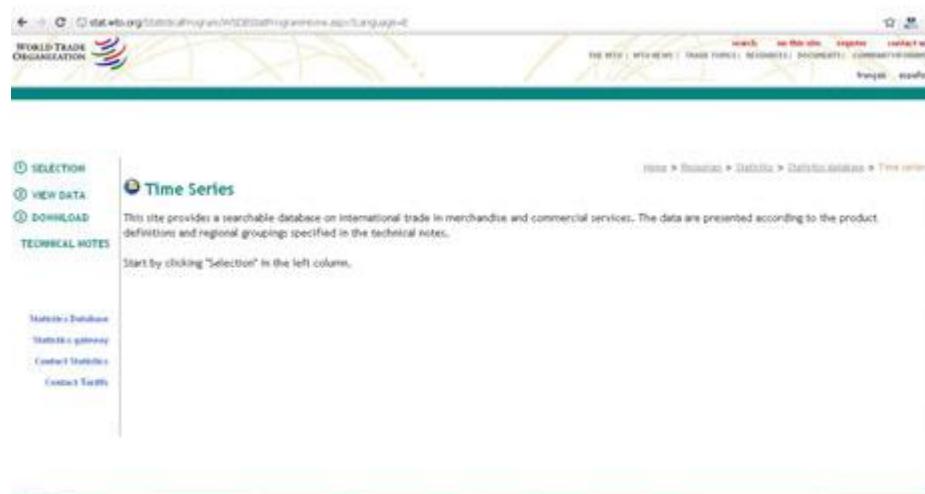
- [1] TS Nguyễn Nam Hải, Giáo trình An toàn cơ sở dữ liệu, Ban Cơ yếu Chính phủ, Học viện Kỹ thuật Mật mã, 2006.

- [2] TS. Nguyễn Đình Vinh, Giáo trình Cơ sở an toàn thông tin, Ban Cơ yếu Chính phủ, Học viện Kỹ thuật Mật mã, 2006.
- [3] Alvaro I. Gálvez, *Multilevel Database Security*, Term Paper, 9- 2013.
- [4] Alfred Basta, Melissa Zgola, *Database Security*, Information Security Professionals, Course technology Cengage Learning, 2012.
- [5]. Meg Coffin Murray Kennesaw State University, Kennesaw, GA, USA —*Database Security: What Students Need to Know* || Journal of Information Technology Education: Innovations in Practice Volume 9, 2010.
- [6] GÜNTHER PERNUL, Institute of Informatics – Information system, University Wien Vienna, Austrilia, Database Security, Advances in Computers, Vol. 38. M. C. Yovits (Ed.), Academic Press, 1994, pp. 1 – 74
- [7] Ravi S. Sandhu and Sushil Jajodia, Center for Secure Information System, *Data and Database Security and Controls*, Handbook of Information Security Management, Auerbach Publishers, 1993, pages 481-499.
- [8] GÜNTHER PERNUL, *Database Security (part 1)*, IPICS 99.
- [9] Elisa Bertino, Fellow, IEEE, and Ravi Sandhu, Fellow, IEEE —*Database Security—Concepts, Approaches, and Challenges* IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL. 2, NO. 1, JANUARY-MARCH 2005.
- [9] O. SamySayadjari, —*Multilevel Security: Reprise*,|| IEEE Security and Privacy, vol. 3, no. 5, 2004.

PHỤ LỤC: Tìm hiểu cơ sở dữ liệu thống kê của Tổ chức Thương mại thế giới (WTO)

Trang web của Tổ chức Thương mại thế giới (WTO) đã xây dựng cơ sở dữ liệu thống kê theo thời gian (Time Series on international trade), người dùng tùy theo nhu cầu, có thể truy nhập vào trang web <http://wto.org> để tra cứu cơ sở dữ liệu về thương mại hàng hóa và dịch vụ quốc tế.

Các dữ liệu được trình bày theo sản phẩm và nhóm khu vực. Chia thành 6 bộ dữ liệu: các mặt hàng (Merchandise trade by commodity), các chỉ số thương mại (Merchandise trade indices), mạng lưới thương mại thế giới (Network of world merchandise trade), các hiệp định hội nhập được chọn (Selected regional integration agreements), tổng thương mại hàng hóa (Total merchandise trade), các dịch vụ thương mại (Trade in commercial services).



Website của WTO

Có thể tra cứu được dữ liệu về xuất khẩu, nhập khẩu, tái nhập khẩu của từng mặt hàng giao thương của mỗi quốc gia với các nước đối tác.

Các bước thực hiện tra cứu số liệu thống kê chuỗi thời gian trên trang web <http://wto.org> được thực hiện như sau:

Bước 1: Truy nhập vào Chuyên mục “[Documents and resources](#)”. (để xem tổng hợp các đề mục) và trỏ chuột vào mục “Documents and resources” để xuất hiện 1 khung danh sách lựa chọn, chọn mục “[Statistics Database](#)”.

ws and ents	Trade topics	WTO membership	Documents and resources
Official documents	Statistics		Multimedia
WTO “Documents Online”	Statistics database		Audio/podcasting
Legal texts	Tariff data		Video
GATT documents	International trade statistics		Photos
	Maps		
Publications	“Made in the World” initiative		Other resources
Key publications	Economic research		Glossary
By title			Distance learning
By category/subject	Working papers		Terminology database
Online bookshop	WTO Chairs Programme		
WTO bookshop in Geneva			
Library			

Bước 2: Vào mục “Time Series on international trade”

The screenshot shows the "Statistics database" section of the WTO website. On the left, there's a sidebar with links to "Statistics gateway", "Contact Statistics", "Contact Tariffs", and a "Copyright" section which includes a note about making copies of the site. The main content area has a heading "Statistics database" with a sub-headline: "Welcome to the WTO statistics database, which allows you to retrieve statistical information from the following presentations." Below this is a bulleted list of features: "Trade Profiles", "Tariff Profiles", "Services Profiles", and "Time Series section". At the bottom of the page, there are links for "Trade Profiles", "Tariff Profiles", "Services Profiles", and "Time Series on international trade".

Bước 3: Lựa chọn các chỉ tiêu trong từng bộ dữ liệu theo các thao tác sau:

Click chuột vào menu “SELECTION” phía trái màn hình, xuất hiện cửa sổ mục chủ đề (Subject)

① SELECTION

② VIEW DATA

③ DOWNLOAD

TECHNICAL NOTES

Statistics Database

Statistics gateway

Time Series - Subject selection

Select one of the data sets available from the dropdown list. Then, select corresponding indicator(s).

AVAILABLE DATA SETS

Merchandise trade by commodity

Exports and imports of major commodity groups by region and selected economy, 1980-2010.

AVAILABLE INDICATORS

Agricultural products

.Food

Fuels and mining products

.Fuels

Manufactures

.Iron and steel

.Chemicals

.Pharmaceuticals

.Machinery and transport equipment

.Office and telecom equipment

SELECTED INDICATORS

ADD

REMOVE

ADD ALL

REMOVE ALL

Click chuột vào mũi tên sổ xuống danh sách các bộ dữ liệu, chọn 1 bộ dữ liệu cần tìm.

Ví dụ: Cần tra cứu cơ sở dữ liệu về thương mại hàng hóa và dịch vụ quốc tế theo các mặt hàng thì chọn “Merchandise trade by commodity”.

Time Series - Subject selection

Select one of the data sets available from the dropdown list. Then, select corresponding indicator(s)

AVAILABLE DATA SETS

Merchandise trade by commodity

Merchandise trade by commodity

Exports and imports of major commodity groups by region and selected economy, 1980-2010.

AVAILABLE INDICATORS

Agricultural products

.Food

Fuels and mining products

.Fuels

Manufactures

.Iron and steel

.Chemicals

.Pharmaceuticals

.Machinery and transport equipment

.Office and telecom equipment

SELECTED INDICATORS

ADD

REMOVE

ADD ALL

REMOVE ALL

Tiếp tục kéo thanh trượt để chọn 1 hoặc nhiều chỉ tiêu cần tìm ở cột bên trái, click chuột vào nút “ADD” để xác nhận.

Ví dụ: Cần tra cứu về mặt hàng thực phẩm (food), nhiên liệu (fuels), và tất cả mặt hàng của ngành công nghiệp chế tạo (manufactures)

Time Series - Subject selection

Select one of the data sets available from the dropdown list. Then, select corresponding indicat

AVAILABLE DATA SETS
Merchandise trade by commodity

Exports and imports of major commodity groups by region and selected economy, 1980-20

AVAILABLE INDICATORS	SELECTED INDICATORS
<ul style="list-style-type: none"> Agricultural products .Food Fuels and mining products .Fuels Manufactures .Iron and steel .Chemicals ..Pharmaceuticals ..Machinery and transport equipment ..Office and telecom equipment 	<ul style="list-style-type: none"> ADD REMOVE ADD ALL REMOVE ALL

Time Series - Subject selection

Select one of the data sets available from the dropdown list. Then, select corresponding indicat

AVAILABLE DATA SETS
Merchandise trade by commodity

Exports and imports of major commodity groups by region and selected economy, 1980-20

AVAILABLE INDICATORS	SELECTED INDICATORS
<ul style="list-style-type: none"> Agricultural products Fuels and mining products .Iron and steel .Chemicals ..Pharmaceuticals ..Machinery and transport equipment ..Office and telecom equipment ...Electronic data processing and office equipment ...Telecommunications equipment ...Integrated circuits and electronic components 	<ul style="list-style-type: none"> .Food .Fuels Manufactures

- Bấm vào nút “continue” (tiếp tục) bên góc phải phía dưới màn hình để tiếp tục lựa chọn cho mục Báo cáo (Reporter)

Tại đây có thể chọn báo cáo của nhóm quốc gia hoặc của riêng mỗi quốc gia. Thực hiện chọn và bấm “ADD” như trên.

Ví dụ: Tra cứu dữ liệu của 3 quốc gia Iraq (Iraq), Ai Len (Ireland), Nhật Bản (Japan).

Time Series - Reporter selection

You can select either individual reporter(s) or group(s) of reporters. Refer to the technique

GROUP
European Union (27)

AVAILABLE REPORTERS
Iceland
India
Indonesia
Iran, Islamic Rep. of
Israel
Italy
Jamaica
Jordan
Kazakhstan
Kenya

SELECTED REPORTERS
Iraq
Ireland
Japan

ADD **REMOVE**
ADD ALL **REMOVE ALL**

- Bấm “continue” tiếp tục lựa chọn cho mục Đối tác (Partner) với thao tác tương tự. Nếu mục Đối tác (Partner) không xuất hiện tức là đã mặc định đối tác là thế giới (World).

- Bấm “continue” tiếp tục lựa chọn cho mục Dòng thương mại (Trade flow), gồm 3 lựa chọn Xuất khẩu (Exports), Nhập khẩu (Imports), Tái xuất khẩu (Re-exports).

Ví dụ: Nếu chọn Xuất khẩu

Time Series - Trade flow selection

Re-exports, when available for selection, are valid only for Singapore and Hong-Kong, China.

AVAILABLE TRADE FLOWS
Imports
Re-exports

SELECTED TRADE FLOWS
Exports

ADD **REMOVE**
ADD ALL **REMOVE ALL**

- Bấm “continue” tiếp tục lựa chọn cho mục Chuỗi thời gian theo năm (Time series – Year)

Ví dụ: Lấy dữ liệu từ năm 1999 - 2010

Time Series - Year selection

AVAILABLE YEARS		SELECTED YEARS	
1998		2010	
1997		2009	
1996		2008	
1995		2007	
1994		2006	
1993		2005	
1992		2004	
1991		2003	
1990		2002	
1989		2001	

ADD
REMOVE
ADD ALL
REMOVE ALL

Bấm “continue” hoặc “VIEW DATA” để xem kết quả cuối cùng của cả 3 nước Iraq, Ai Len, Nhật Bản.

- Bấm vào “Year” để xem những năm khác

Bấm vào “View printable version” để xem kết quả ở định dạng

Subject: Merchandise trade by commodity Unit: US dollar at current prices (Millions)

Reporter	Flow	Indicator	Partner	2006	2007	2008	2009	2010
Iraq	Exports	Food	World	52	42	30	13	26
Iraq	Exports	Fuels	World	29202	41115	62148	39407	52673
Iraq	Exports	Manufactures	World	93	106	4	10	2
Ireland	Exports	Food	World	10566	11968	11823	9961	10553
Ireland	Exports	Fuels	World	706	941	1224	830	1150
Ireland	Exports	Manufactures	World	91464	102490	106577	99077	98597
Japan	Exports	Food	World	3147	3623	3980	4010	4755
Japan	Exports	Fuels	World	5897	9280	18776	10530	13038
Japan	Exports	Manufactures	World	586521	640873	693235	507992	680290