STUDIA INFORMATICA 2004
Volume 25 Number 1 (57)

Marcin GORAWSKI, Jacek FRĄCZEK
Politechnika Śląska, Instytut Informatyki

IMPLEMENTACJA PROCEDUR BEZPIECZEŃSTWA W HURTOWNIACH DANYCH

Streszczenie. Praca opisuje proces implementacji systemu zabezpieczeń kontrolującego dostęp do szczegółowych informacji zawartych w hurtowni danych typu ROLAP. Na przykładzie *Systemu Analizy Zasobów Ludzkich* przedstawiono praktyczną realizację złożonych procedur bezpieczeństwa. Opisano i porównano dwie metody implementacji systemu zabezpieczeń – z wykorzystaniem perspektyw i wirtualnych, prywatnych baz danych.

Słowa kluczowe: hurtownia danych, system bezpieczeństwa, kontrola dostępu na poziomie wierszy

THE IMPLEMENTATION OF SECURITY POLICIES IN DATA WAREHOUSES

Summary. This research describes the process of implementation of a security system that controls access to detail information in a ROLAP data warehouse. The *Human Resources Analysis System* is presented as an example for a practical implementation of complex security policies. Two methods of implementation of the security system are described and discussed: database views and virtual private databases.

Keywords: data warehouse, security system, row-level access control

1. Wstęp

Jednym z kluczowych etapów projektu systemu bazodanowego jest ustalenie reguł dostępu użytkowników do danych. Przyjęcie odpowiedniej polityki bezpieczeństwa i sposób jej implementacji są szczególnie ważne w przypadku systemów hurtowni danych [2]. Integracja danych pochodzących z różnych podsystemów informatycznych przedsiębiorstwa,

wysoka jakość gromadzonej informacji i jej ukierunkowanie na podejmowanie strategicznych decyzji zarządczych decydują o wysokiej wartości danych przechowywanych w hurtowni. Udostępnianie tego typu danych szerszym grupom użytkowników wymaga zapewnienia szczególnej ochrony informacji [1]: jej poufności [7], integralności [5] i dostępności [9].

W pracy [3] szczegółowo opisano charakterystyczne dla hurtowni danych modele kontroli dostępu do informacji oraz wskazano mechanizmy ich implementacji. Na sposób implementacji reguł polityki bezpieczeństwa w największym stopniu wpływają:

- przyjęta architektura systemu – a w szczególności rodzaj komponentu, który jest odpowiedzialny za realizację kontroli dostępu (baza danych, serwer MOLAP, aplikacja OLAP lub rozwiązanie bazujące na współpracy wymienionych elementów) [8],
- dostępne w systemie mechanizmy realizacji zabezpieczeń,
- złożoność przyjętej polityki bezpieczeństwa,
- wymagania związane z wydajnością pracy zabezpieczanego systemu.

Szczególnie szerokie możliwości implementacji mechanizmów bezpieczeństwa istnieją w hurtowniach danych typu ROLAP (ang. *Relational Online Analytical Processing*), które przechowują informacje w relacyjnych bazach danych i mają możliwość realizacji kontroli dostępu na poziomie systemu zarządzania bazą danych. W pracy [3] dokonano analizy mechanizmów zabezpieczeń dostępnych w hurtowniach tego typu z uwzględnieniem kontroli dostępu na poziomach: systemu bazy danych, poszczególnych obiektów bazy danych oraz na poziomie kolumn i wierszy danych.

Niniejsza praca opisuje praktyczną realizację systemu bezpieczeństwa dla hurtowni typu ROLAP, który został stworzony w oparciu o stosunkowo złożony zestaw reguł. Prezentowany w pracy *System Analizy Zasobów Ludzkich* jest przykładem rzeczywistego projektu hurtowni danych. Założenia przyjęte przy budowie opisywanego systemu zabezpieczeń i sposób ich realizacji są w dużej części uniwersalne i można je wykorzystać przy tworzeniu innych projektów. W celu bardziej obrazowego ukazania procesu wdrożenia systemu bezpieczeństwa procedurę implementacji podzielono na mniejsze etapy.

2. System Analizy Zasobów Ludzkich

Przedstawiany *System Analizy Zasobów Ludzkich* jest specjalizowanym podsystemem hurtowni danych przeznaczonym do analizy indywidualnych i zbiorczych cech kadry przedsiębiorstwa (wiek, płeć, wykształcenie i inne) oraz raportowania poziomu zatrudnienia i frekwencji pracowników.

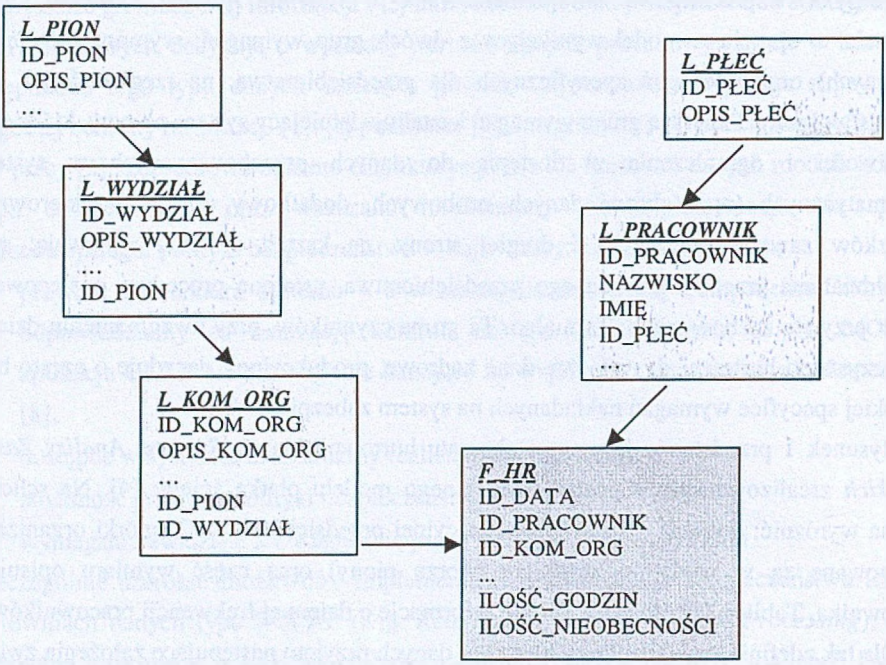
Polityka bezpieczeństwa zaimplementowana w *Systemie Analizy Zasobów Ludzkich* powstała w oparciu o model wynikający z dwóch grup wymagań: wymogów formalnych (prawnych) oraz wymagań specyficznych dla przedsiębiorstwa, na rzecz którego system został stworzony. Pierwszą grupę wymagań kształtuje istniejący system prawny, który określa możliwości i ograniczenia w dostępie do danych przechowywanych w systemach informatycznych (np. ochrona danych osobowych, dodatkowy dostęp dla kierownictwa związków zawodowych itp.). Z drugiej strony, na kształt systemu wpływają: sposób współdziałania pracowników danego przedsiębiorstwa, ustalone procedury postępowania a nawet przyjęte zachowania nieformalne. Ta grupa czynników, przy uwzględnieniu dziedziny wykorzystania hurtowni danych (np. dane kadrowe, produkcyjne), decyduje o często bardzo wysokiej specyfice wymagań nakładanych na system zabezpieczeń.

Rysunek 1 przedstawia fragment schematu hurtowni danych *Systemu Analizy Zasobów Ludzkich* zrealizowanego w postaci klasycznego modelu płatka śniegu [4]. Na schemacie można wyróżnić: wymiar struktury organizacyjnej przedsiębiorstwa (komórki organizacyjne zgrupowane są w wydziały, wydziały tworzą piony) oraz część wymiaru opisującego pracownika. Tablica faktów przechowuje informacje o dziennej frekwencji pracowników.

Dla tak zdefiniowanej struktury hurtowni danych przyjęto następujące założenia związane z wdrażaniem systemu kontroli dostępu do danych:

- pracownicy uzyskują dostęp do danych na podstawie przydzielonych im praw,
- system będzie wdrażany etapowo:
  - etap I – kadra kierownicza uzyskuje możliwość swobodnego przeglądania całej struktury organizacyjnej przedsiębiorstwa (pionów, wydziałów, komórek), ale posiada dostęp do danych (faktów) tylko tych obiektów struktury, do których jawnie nadano jej prawa, np. kierownik wydziału powinien mieć możliwość przeglądania danych wszystkich pracowników podległych mu komórek organizacyjnych,
  - etap II – pracownik uzyskuje prawo wglądu do swoich danych osobowych (imię, nazwisko, data urodzenia, PESEL) zawartych w wymiarze opisującym pracownika,
  - etap III – pracownik uzyskuje dostęp do danych o swojej frekwencji (dane znajdują się w tablicy faktów),
  - etap IV – wdrożenie ochrony danych osobowych i udostępnienie informacji szerszemu gronu użytkowników.

Przyjęte założenia wymuszają realizację mieszanego (hierarchiczno-indywidualnego, [3]) modelu bezpieczeństwa opartego na tablicy praw i perspektywach bezpieczeństwa [6,10] lub też wykorzystującego prywatną, wirtualną bazę danych [12].



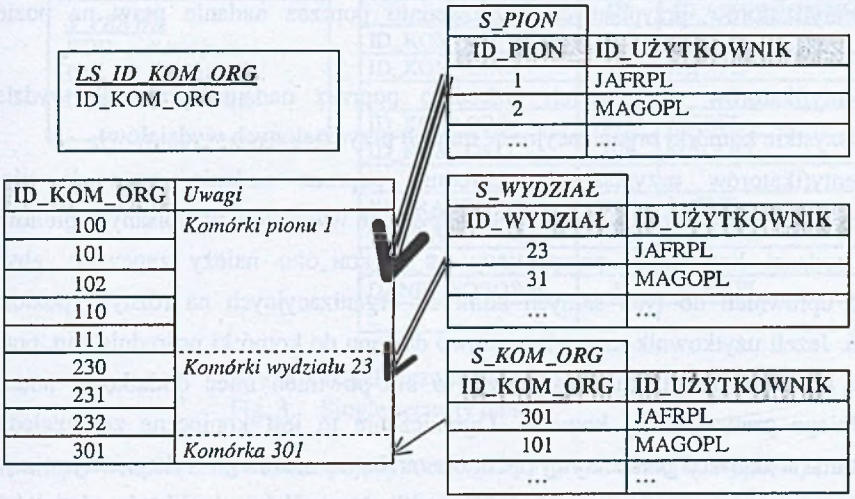
Rys. 1. Fragment schematu omawianej hurtowni danych
Fig. 1. Fragment of schema of the discussed data warehouse

2.1. Etap I - autoryzacja według struktury organizacyjnej

Etap I zakłada udostępnienie użytkownikom danych według posiadanych przez nich praw do komórek struktury organizacyjnej przedsiębiorstwa. Użytkownikom wyższych szczebli kierowniczych nadawany jest dostęp do wszystkich komórek swojego pionu(ów) lub wydziału(ów), kierownikom komórek nadaje się dostęp wyłącznie do swojej komórki organizacyjnej, a pozostałym użytkownikom nie przydziela się uprawnień.

W celu realizacji odpowiednich mechanizmów zabezpieczających z tablicami wymiaru struktury organizacyjnej (rys.1) kojarzy się tablice praw dostępu (rys.2 – `S_PION`, `S_WYDZIAŁ`, `S_KOM_ORG`), które – dla danego użytkownika – przechowują identyfikatory udostępnionych mu danych. Przypisanie każdej zabezpieczonej klasie obiektów pojedynczej tablicy praw pozwala na łatwe ustalenie zakresu danych, do których użytkownik ma dostęp.

W celu zabezpieczenia dostępu do danych według hierarchii organizacyjnej, na podstawie utworzonych wcześniej tablic praw dostępu, tworzy się perspektywy `LS_ID_KOM_ORG` oraz `FS_HR_1`.



Rys. 2. Tablice praw dostępu i widok zabezpieczający hierarchię organizacyjną (dla użytkownika *jafrpl*)

Fig. 2. Security tables and the security view for an organizational hierarchy (for user *jafrpl*)

Perspektywa `LS_ID_KOM_ORG` umożliwia wyznaczenie pełnego zestawu identyfikatorów komórek organizacyjnych, do których ma dostęp dany użytkownik (podana składnia definicji perspektywy `LS_ID_KOM_ORG` i innych obiektów przedstawionych w pracy jest charakterystyczna dla bazy danych Oracle):

```
CREATE OR REPLACE VIEW LS_ID_KOM_ORG
AS
        SELECT ID_KOM_ORG
        FROM S_KOM_ORG
        WHERE ID_UZYTKOWNIK = USER
UNION ALL
        SELECT k.ID_KOM_ORG
        FROM S_WYDZIAL w, L_KOM_ORG k
        WHERE k.ID_WYDZIAL = w.ID_WYDZIAL
                AND w.ID_UZYTKOWNIK = USER
UNION ALL
        SELECT k.ID_KOM_ORG
        FROM S_PION p, L_KOM_ORG k
        WHERE k.ID_PION = p.ID_PION
                AND p.ID_UZYTKOWNIK = USER;
```

W celu identyfikacji użytkownika pracującego z perspektywą wykorzystuje się funkcję `USER`, zwracającą nazwę użytkownika bieżącej sesji. Wynikowy zestaw identyfikatorów komórek organizacyjnych udostępnionych danemu użytkownikowi uzyskuje się poprzez zsumowanie zbiorów:

- identyfikatorów przypisanych bezpośrednio poprzez nadanie praw na poziomie komórek organizacyjnych,
- identyfikatorów przypisanych pośrednio poprzez nadanie praw do wydziałów (wszystkie komórki organizacyjne w ramach przydzielonych wydziałów),
- identyfikatorów przypisanych pośrednio poprzez nadanie praw do pionów przedsiębiorstwa (wszystkie komórki wszystkich wydziałów przypisanych pionów).

Przy podanej konstrukcji perspektywy LS_ID_KOM_ORG należy zapewnić, aby nie powtarzać uprawnień do tych samych komórek organizacyjnych na różnych poziomach uprawnień. Jeżeli użytkownik ma nadane prawo dostępu do komórki pośrednio (np. poprzez prawo do nadrzędnego dla niej wydziału), to nie powinien mieć dodatkowo nadanego bezpośredniego prawa do tej komórki. Ograniczenie to jest konieczne ze względu na wykorzystanie w definicji perspektywy operatora UNION ALL, którego wykonanie jest szybsze od operatora UNION, a który nie usuwa ze zbioru wyników duplikatów (wielokrotne pojawienie się tych samych danych będzie prowadziło do błędnego działania aplikacji OLAP). Dodatkowo, przy realizacji podanych przykładów, należy zapewnić posiadanie przez każdego z użytkowników odpowiedniego zestawu praw systemowych i praw dostępu do zdefiniowanych wyżej obiektów.

W oparciu o perspektywę LS_ID_KOM_ORG budujemy widok FS_HR_1 zabezpieczający tablicę bazową:

```
CREATE OR REPLACE VIEW FS_HR_1
AS
SELECT f.ID_DATA, f.ID_PRACOWNIK, f.ID_KOM_ORG,
       f.ILOSC_GODZIN, f.ILOSC_NIEOBECNOSCI
FROM F_HR f, LS_ID_KOM_ORG s
WHERE f.ID_KOM_ORG = s.ID_KOM_ORG;
```

Dążąc do uproszczenia i przyspieszenia działania systemu autoryzacji można zbudować pojedynczą tablicę praw dostępu, która byłaby odpowiedzialna za przechowywanie identyfikatorów obiektów różnych klas (rys.3). Dodatkowo można zmodyfikować sposób nadawania praw do danych według struktury organizacyjnej tak, aby uniknąć wielokrotnego wyznaczania (dla każdego zapytania) zestawu identyfikatorów komórek organizacyjnych związanych z posiadanymi prawami do pionów i wydziałów. W tym celu w momencie nadania prawa na wyższym poziomie hierarchii należy wyznaczyć wszystkie przypisane do niego komórki organizacyjne i taki zbiór zapisać na stałe w tablicy praw dostępu. Rozwiązanie to wymaga bardziej złożonej aplikacji służącej do zarządzania prawami, która z jednej strony powinna umożliwiać wizualizację praw nadanych na wyższych poziomach, a z drugiej strony powinna operować wyłącznie na prawach nadawanych na najniższym poziomie hierarchii.

S_PRAWA	
KOD	*// kod prawa*
ID	*// id atrybutu*
ID_UŻYTKOWNIK	

KOD	ID	ID_UZYTKOWNIK
ID_KOM_ORG	100	JAFRPL
ID_KOM_ORG	101	JAFRPL
...	...	...
ID_KOM_ORG	301	JAFRPL
ID_KOM_ORG	101	MAGOPL
...	...	...
ID_PRACOWNIK	1001	JAFRPL
ID_PRACOWNIK	1002	MAGOPL
...	...	...
DANE OSOBOWE	1	DOWAPL
...	...	...

Rys. 3. Pojedyncza tablica praw dostępu
Fig. 3. Single security table

Struktury bezpieczeństwa dla pojedynczej tablicy praw dostępu wyglądają następująco (perspektywa `LS_ID_KOM_ORG` nie jest konieczna):

```
CREATE OR REPLACE VIEW FS_HR_2
AS
SELECT f.ID_DATA, f.ID_PRACOWNIK, f.ID_KOM_ORG,
       ILOSC_GODZIN, ILOSC_NIEOBECNOSCI
FROM F_HR f, S_PRAWA s
WHERE
       f.ID_KOM_ORG = s.ID
       AND ID_UZYTKOWNIK = USER
       AND KOD = 'ID_KOM_ORG';
```

Związane z etapem I wymaganie zapewnienia możliwości nieskrępowanego przeglądania pełnej struktury organizacyjnej przedsiębiorstwa (tablice `L_PION`, `L_WYDZIAL`, `L_KOM_ORG`) pozwala uniknąć zabezpieczenia tablic tego wymiaru, przez co proces przeglądania będzie wykonywany z pełną i stałą dla wszystkich pracowników wydajnością.

2.2. Etap II – dostęp do informacji osobowych

Umożliwienie dostępu pracownika do swoich danych osobowych (imię, nazwisko, data urodzenia, `PESEL`) zawartych w wymiarze hurtowni danych jest przykładem zabezpieczenia informacji pochodzącej z tablicy wymiaru schematu hurtowni.

Odpowiednie prawa wiążące identyfikator danych użytkownika w hurtowni (`ID_PRACOWNIK` w tablicy `L_PRACOWNIK`) z jego nazwą (`ID_UZYTKOWNIK`) przechowywane są – podobnie jak w etapie I – w tablicy praw `S_PRAWA` (kod uprawnień *`ID_PRACOWNIK`*). Perspektywa zabezpieczająca tablicę `L_PRACOWNIK` ma następującą definicję:

```
CREATE OR REPLACE VIEW LS_PRACOWNIK_2
AS
SELECT p.ID_PRACOWNIK, NAZWISKO, IMIE, ID_PLEC, PESEL, DATA_UR
FROM S_PRAWA s, L_PRACOWNIK p
WHERE s.KOD = 'ID_PRACOWNIK'
      AND s.ID = p.ID_PRACOWNIK
      AND S.ID_UZYTKOWNIK = USER;
```

W celu realizacji wymagań etapu II każdy z użytkowników powinien mieć w tablicy S_PRAWA wpis umożliwiający dostęp do własnych danych.

Podobne zabezpieczenie można uzyskać poprzez uzupełnienie danych pracownika (w tablicy LS_PRACOWNIK) o jego identyfikator w systemie komputerowym (kolumna ID_UZYTKOWNIK) i bezpośrednie odwołanie się w definicji perspektywy zabezpieczającej do tego pola:

```
CREATE OR REPLACE VIEW LS_PRACOWNIK_2A
AS
SELECT ID_PRACOWNIK, NAZWISKO, IMIE, ID_PLEC, PESEL, DATA_UR
FROM L_PRACOWNIK
WHERE ID_UZYTKOWNIK = USER;
```

Perspektywa LS_PRACOWNIK_2A ma prostszą definicję i działa szybciej od perspektywy LS_PRACOWNIK_2, ma natomiast mniejszą funkcjonalność. Perspektywa LS_PRACOWNIK_2A pozwala bowiem na dotarcie do informacji tylko o sobie samym, natomiast w przypadku perspektywy LS_PRACOWNIK_2 mamy możliwość udostępnienia informacji także o innych osobach. Efekt taki uzyskuje się poprzez przypisanie danemu pracownikowi w tablicy S_PRAWA większej ilości praw typu ID_PRACOWNIK.

2.3. Etap III – indywidualny dostęp do danych Systemu Analizy Zasobów Ludzkich

Etap III rozszerza możliwości dostępu do danych tablicy faktów F_HR, pozwalając użytkownikom na analizę danych osobistych – niezależnie od posiadanych uprawnień na poziomie struktury organizacyjnej. Podane wymaganie można zrealizować poprzez budowę nowej perspektywy zabezpieczającej FS_OSOBOWY_HR lub też poprzez rozbudowę definicji perspektywy zabezpieczającej tablicę faktów (FS_HR_3).

Nową perspektywę FS_OSOBOWY_HR można zdefiniować w następujący sposób:

```
CREATE OR REPLACE VIEW FS_OSOBOWY_HR
AS
SELECT f.ID_DATA, f.ID_PRACOWNIK, f.ID_KOM_ORG,
       ILOSC_GODZIN, ILOSC_NIEOBECNOSCI
FROM F_HR f, S_PRAWA s
WHERE
       f.ID_PRACOWNIK = s.ID
       AND ID_UZYTKOWNIK = USER
       AND KOD = 'ID_PRACOWNIK';
```

Rozszerzona perspektywa FS_HR_3 ma budowę:

```
CREATE OR REPLACE VIEW FS_HR_3
AS
        SELECT f.ID_DATA, f.ID_PRACOWNIK, f.ID_KOM_ORG,
               ILOSC_GODZIN, ILOSC_NIEOBECNOSCI
        FROM F_HR f, S_PRAWA s
        WHERE
               f.ID_KOM_ORG = s.ID
               AND ID_UZYTKOWNIK = USER
               AND KOD = 'ID_KOM_ORG'
UNION
        SELECT f.ID_DATA, f.ID_PRACOWNIK, f.ID_KOM_ORG,
               ILOSC_GODZIN, ILOSC_NIEOBECNOSCI
        FROM F_HR f, S_PRAWA s
        WHERE
               f.ID_PRACOWNIK = s.ID
               AND ID_UZYTKOWNIK = USER
               AND KOD = 'ID_PRACOWNIK';
```

W przypadku perspektywy FS_HR_3 należy zwrócić uwagę na podwójne złączenie tablicy bazowej F_HR z tablicą praw S_PRAWA – według kolumn F_HR.ID_KOM_ORG oraz F_HR.ID_PRACOWNIK. W celu uniknięcia podwójnego zliczania wierszy w definicji perspektywy użyto frazy UNION. Można też wykorzystać konstrukcję z zapytaniem zagnieżdżonym (którego plan wykonania może być bardziej efektywny):

```
CREATE OR REPLACE VIEW FS_HR_3A
AS
        SELECT ID_DATA, ID_PRACOWNIK, ID_KOM_ORG,
               ILOSC_GODZIN, ILOSC_NIEOBECNOSCI
        FROM F_HR
        WHERE
               ID_KOM_ORG IN (
                  SELECT ID FROM S_PRAWA
                  WHERE
                     ID_UZYTKOWNIK = USER
                     AND KOD = 'ID_KOM_ORG')
               OR ID_PRACOWNIK IN (
                  SELECT ID FROM S_PRAWA
                  WHERE
                     ID_UZYTKOWNIK = USER
                     AND KOD = 'ID_PRACOWNIK');
```

2.4. Etap IV – wdrożenie ochrony danych osobowych

Wdrożenie etapu IV stanowi przykład realizacji złożonego wymagania polegającego na ukryciu części danych opisujących pracownika (m.in.: nazwiska, imienia, numeru PESEL),

przy jednoczesnym zapewnieniu możliwości wykonywania raportów korzystających z pozostałych danych tego wymiaru (np. raporty uwzględniające płeć pracownika).

Postawiony problem można w najprostszy sposób rozwiązać poprzez nadanie praw selekcji tylko do udostępnionych kolumn tablicy LS_PRACOWNIK (lub widoku zabezpieczającego tę tablicę). Niestety, komercyjne aplikacje OLAP z reguły nie potrafią obsłużyć raportów, które dla jednego użytkownika zwracałyby dane z wszystkich kolumn, a dla innego tylko z części (zazwyczaj generowane jest polecenie SELECT *, które w takim przypadku zakończy się błędem braku dostępu). Podobnie, nie jest dobrym rozwiązaniem utworzenie prostej perspektywy udostępniającej wybrane kolumny tablicy L_PRACOWNIK bez kolumn danych osobowych:

```
CREATE OR REPLACE VIEW LS_PRACOWNIK_3
AS
        SELECT ID_PRACOWNIK, ID_PLEC          -- tylko „dozwolone" kolumny
        FROM L_PRACOWNIK;
```

Struktura perspektywy LS_PRACOWNIK_3 jest zasadniczo różna od struktury tablicy L_PRACOWNIK. Z tego względu koniecznie jest istnienie dwóch obiektów: tablicy L_PRACOWNIK dla osób z dostępem do danych osobowych i perspektywy LS_PRACOWNIK_3 dla pozostałych użytkowników. W przypadku używania narzędzi OLAP ten typ rozwiązania może prowadzić do utworzenia i utrzymywania 2 niezależnych aplikacji, co jest zjawiskiem niepożądanym.

W celu utworzenia perspektywy zabezpieczającej o strukturze tablicy użytej w definicji tej perspektywy można wykorzystać procedury (funkcje) składowane. Przedstawiona poniżej funkcja CHECK_HR_RIGHTS sprawdza, czy dana osoba posiada w tablicy S_PRAWA odpowiednie prawo (o kodzie *DANE OSOBOWE*) związane z dostępem do danych osobowych. Z funkcji tej w dalszej kolejności korzystają procedury sterujące sposobem wyświetlania danych określonego typu (S_VARCHAR2 i analogiczne: S_NUMBER, S_DATE). Osoby, które mogą przeglądać dane osobowe, otrzymują informacje rzeczywiste, natomiast osobom pozostałym zwracane są wartości NULL.

```
CREATE OR REPLACE FUNCTION CHECK_HR_RIGHTS RETURN BOOLEAN
IS
        cursor check_rights is
                select * from S_PRAWA
                        where KOD = 'DANE OSOBOWE'
                        and ID_UZYTKOWNIK = USER;
        rec check_rights%ROWTYPE;

BEGIN
        open check_rights;
        fetch check_rights into rec;
        if check_rights%NOTFOUND then
                close check_rights;
```

```
            return FALSE;
        end if;
        close check_rights;
        return TRUE;
END;

CREATE OR REPLACE FUNCTION S_VARCHAR2(PAR VARCHAR2) RETURN VARCHAR2
IS
BEGIN
        if check_HR_rights then return par;
        else return NULL;
        end if;
END;
```

Perspektywa zabezpieczająca tablicę `L_PRACOWNIK` z wykorzystaniem funkcji składowanych ma budowę:

```
CREATE OR REPLACE VIEW LS_PRACOWNIK_4
AS
SELECT ID_PRACOWNIK, s_varchar2(NAZWISKO) Nazwisko,
        s_varchar2(IMIE) Imie, ID_PLEC, s_number(PESEL) PESEL,
        s_date(DATA_UR) data_ur
FROM L_PRACOWNIK;
```

Podane rozwiązanie wymaga znacznej mocy obliczeniowej serwera bazy danych. Bardziej wydajną implementację opisanego wymagania można uzyskać definiując odpowiednie zabezpieczenia związane z wyświetlaniem danych na poziomie aplikacji (z tym, że nie wszystkie narzędzia OLAP umożliwiają uzyskanie żądanego efektu) lub też korzystając z możliwości dynamicznego tworzenia obiektów bazy danych przy uruchamianiu aplikacji. Jeżeli możliwe jest dynamiczne tworzenie obiektów, to dla użytkownika, który nie posiada praw do przeglądania danych osobowych, można utworzyć perspektywę `L_PRACOWNIK_5`:

```
CREATE OR REPLACE VIEW LS_PRACOWNIK_5
AS
SELECT ID_PRACOWNIK, '' as Nazwisko, '' as Imie,
        ID_PLEC, '' as PESEL, '' as data_ur
FROM L_PRACOWNIK;
```

2.5. Wykorzystanie prywatnych, wirtualnych baz danych

Prywatne, wirtualne bazy danych – dostępne m.in. w Oracle 8/9 [12] – pozwalają na stosunkowo prostą i bardzo elastyczną implementację złożonych schematów bezpieczeństwa. Zaproponowane rozwiązanie bazuje na mechanizmie automatycznej modyfikacji zapytań użytkownika przez motor bazy danych. Szczególnymi zaletami tego typu rozwiązania jest zachowanie pierwotnego schematu bazy danych oraz możliwość wykorzystania języka programowania przy implementacji algorytmu określającego sposób dostępu do danych. W

przeciwieństwie do rozwiązań prezentowanych wyżej nie jest konieczne tworzenie nowych obiektów (perspektyw) zabezpieczających dostęp do tablic schematu wyjściowego.

W najprostszym przypadku definicję procedur bezpieczeństwa dla prywatnej, wirtualnej bazy danych tworzy się w dwóch krokach [11]:

1 – utworzenie funkcji bezpieczeństwa określającej ograniczenia w dostępie do danych,

2 – aktywacja procedury bezpieczeństwa bazującej na funkcji zdefiniowanej w punkcie 1.

Funkcja bezpieczeństwa odpowiedzialna jest za generację frazy warunku WHERE ograniczającej zakres danych udostępnianych użytkownikowi. Funkcja bezpieczeństwa, która realizowałaby wyżej zdefiniowane etapy I-III może wyglądać następująco:

```
CREATE OR REPLACE FUNCTION SEC_HR
  (P_SCHEMA IN VARCHAR2, P_OBJECT IN VARCHAR2) RETURN VARCHAR2
  AS
  BEGIN
  if (USER='HR_ADM') then                    -- dla administratora
        return '';                           -- brak ograniczeń
  elsif (p_object = 'F_HR') then             -- tablica F_HR
        return 'id_kom_org in (              -- etap I
           select ID from S_PRAWA
           where
              ID_UZYTKOWNIK = SYS_CONTEXT(''USERENV'',''SESSION_USER'')
              and KOD = ''ID_KOM_ORG'')
           OR id_pracownik in (              -- etap III
           select ID from S_PRAWA
           where
              ID_UZYTKOWNIK = SYS_CONTEXT(''USERENV'',''SESSION_USER'')
              AND KOD = ''ID_PRACOWNIK'')';

  elsif (p_object = 'L_PRACOWNIK') then      -- tablica L_PRACOWNIK
        return 'id_pracownik in (            -- etap II
              select ID from S_PRAWA
            where
              ID_UZYTKOWNIK = SYS_CONTEXT(''USERENV'',''SESSION_USER'')
              and KOD = ''ID_PRACOWNIK'' )';
  else
        return '1=2';      -- dla wszystkich pozostałych przypadków
  end if;
END;
```

Występująca w przedstawionym kodzie funkcja SYS_CONTEXT z parametrami USERENV i SESSION_USER pozwala na określenie użytkownika, na rzecz którego jest ona wywoływana (podobnie jak funkcja USER występująca w poprzednich przykładach).

Włączenie procedur bezpieczeństwa ograniczających operacje na tablicach F_HR i L_PRACOWNIK odbywa się za pomocą funkcji pakietu DBMS_RLS:

```
BEGIN
dbms_rls.add_policy
 (object_schema => 'HR_ADM',
   object_name => 'F_HR',
   policy_name => 'Bezp wg kom org',
   function_schema => 'HR_ADM',
   policy_function => 'SEC_HR',
   statement_types => 'select, insert, update, delete',
   update_check => TRUE);

dbms_rls.add_policy
 (object_schema => 'HR_ADM',
   object_name => 'L_PRACOWNIK',
   policy_name => 'Dane wlasne pracownika',
   function_schema => 'HR_ADM',
   policy_function => 'SEC_HR',
   statement_types => 'select, insert, update, delete',
   update_check => TRUE);
END;
```

Po wdrożeniu systemu zabezpieczeń wirtualnej, prywatnej bazy danych użytkownicy (aplikacje) mogą w bezpośredni sposób odwoływać się do tablic hurtowni danych. Polecenia:

```
select * from f_hr;
```

oraz

```
select * from l_pracownik;
```

będą automatycznie zamieniane na polecenia:

```
select * from f_hr
where id_kom_org in (
      select ID from S_PRAWA
      where
         ID_UZYTKOWNIK = SYS_CONTEXT('USERENV','SESSION_USER')
         and KOD = 'ID_KOM_ORG')
OR id_pracownik in (
      select ID from S_PRAWA
      where
         ID_UZYTKOWNIK = SYS_CONTEXT('USERENV','SESSION_USER')
         and KOD = 'ID_PRACOWNIK')';
```

oraz

```
select * from l_pracownik
where id_pracownik in (
      select ID from S_PRAWA
      where
         ID_UZYTKOWNIK = SYS_CONTEXT('USERENV','SESSION_USER')
         and KOD = 'ID_PRACOWNIK');
```

Ze względu na fakt, że wirtualne, prywatne bazy danych umożliwiają realizację mechanizmów kontroli dostępu wyłącznie na poziomie wierszy, realizację etapu IV

(wdrożenie ochrony danych osobowych, które wymaga zabezpieczenia kolumn) należy przeprowadzić w sposób opisany w poprzednim rozdziale[1].

3. Podsumowanie

Wypracowana polityka bezpieczeństwa i sposób jej implementacji w zasadniczy sposób wpływają na kształt i działanie systemu bazodanowego. Wdrożenie mechanizmów bezpieczeństwa z jednej strony pozwala osiągnąć wymaganą funkcjonalność systemu, z drugiej zaś strony wymaga większego wysiłku programistycznego i administracyjnego oraz wpływa na wydajność pracy systemu.

Spośród wielu dostępnych mechanizmów implementacji procedur bezpieczeństwa [3] projektanci powinni wybierać te, które pozwolą na budowę systemu elastycznego, prostego w utrzymaniu i wydajnego. Realizacja tych postulatów jest szczególnie ważna w systemach hurtowni danych, w których złożonym wymogom narzuconym przez procedury bezpieczeństwa towarzyszą duże rozmiary struktur przechowujących dane (m.in. dodatkowej tablicy praw).

Implementacja mechanizmów zabezpieczających z wykorzystaniem tablicy praw dostępu pozwala modyfikować posiadane przez użytkownika uprawnienia bez potrzeby zmian definicji obiektów bazodanowych odpowiedzialnych za zabezpieczenie danych (perspektyw, procedur). Uzyskana elastyczność rozwiązania z reguły wiąże się z obniżeniem wydajności pracy systemu (szczególnie w przypadku dużego rozmiaru tablicy praw). Z tego względu wdrożeniu systemu bezpieczeństwa powinien towarzyszyć proces optymalizacji wydajnościowej, obejmujący m.in.: odpowiednie partycjonowanie i indeksowanie tablic, zbieranie statystyk, analizę planów wykonania zapytań i ewentualną modyfikację ich formy, z uwzględnieniem możliwości równoległej realizacji zapytań. Właściwe wykorzystanie mechanizmów optymalizujących może prowadzić do szybszego wykonywania pewnych zapytań w hurtowni danych z systemem bezpieczeństwa, niż w systemie bez zabezpieczeń. Pojawiające się zwiększenie wydajności można wytłumaczyć nałożeniem dodatkowych warunków selekcji (związanych z systemem bezpieczeństwa), które powodują zmniejszenie rozmiaru wynikowego zbioru danych.

Dla opisanego w pracy *Systemu Analizy Zasobów Ludzkich* przedstawiono dwie realizacje systemu zabezpieczeń. Pierwsza z implementacji wykorzystuje dodatkowe perspektywy zabezpieczające dostęp do tablic hurtowni danych. Metoda ta:

---

[1] W Oracle 10g, w ramach mechanizmu prywatnych, wirtualnych baz danych, wprowadzono możliwość zabezpieczania poszczególnych kolumn tablic.

- powoduje powstanie stosunkowo dużej liczby nowych obiektów schematu bazy danych (po jednym na każdy obiekt zabezpieczany),
- jest kosztowna w utrzymaniu ze względu na rozproszenie logiki odpowiedzialnej za realizację polityki bezpieczeństwa,
- w przypadku złożonych polityk bezpieczeństwa może być trudna w realizacji.

Zaletą metody jest możliwość jej wykorzystania w większości systemów hurtowni danych typu ROLAP. Drugi sposób realizacji wykorzystuje technologię prywatnych, wirtualnych baz danych. Metoda ta:

- zachowuje schemat obiektów bazy danych (tablic, perspektyw),
- pozwala zgromadzić logikę odpowiedzialną za realizację polityki bezpieczeństwa w pojedynczej funkcji (pakiecie funkcji), co zmniejsza koszty utrzymania systemu,
- ze względu na wykorzystanie języka programowania pozwala na implementację złożonych polityk bezpieczeństwa.

LITERATURA

1. Castano S., Fugini M., Matrella G., Samarati P.: Database Security. ACM Press, 1995.
2. Kimball R., Reeves L., Margy R., Thornthwaite W.: The Data Warehouse Lifecycle Toolkit. John Wiley & Sons, 1998.
3. Gorawski M., Frączek J.: Kontrola dostępu do informacji w hurtowniach danych. ZN Pol. Śl., Studia Informatica Vol. 24, No 2B(54), Gliwice 2003.
4. Kimball R.: The Data Warehouse Toolkit. John Wiley & Sons, Inc, 1996.
5. Sandhu R., Jajodia S.: Integrity Mechanisms in Database Management Systems. Proc. 13th National Computer Security Conf., 1990.
6. Rosenthal A, Sciore E.: View Security as the Basis for Data Warehouse Security. Proc. of the Int. Workshop on Design and Management of Data Warehouses. Stockholm, 2000.
7. Sandhu R., Samarati P.: Access Control: Principles and Practice. IEEE Communications, vol.32, no.9, 1994.
8. Priebe T., Penrul G.: Towards OLAP Security Design – Survey and Research Issues. Proc. of the 3rd ACM International Workshop on Data Warehousing and OLAP (DOLAP 2000). Washington, 2000.
9. Babb A. i in.: Maximum Availability Architecture. An Oracle white paper. High Availability System Group. Oracle Corporation, 2003.
10. Administrator, Intelligence Server, and Web Administrator Guide. Version 7.2. Third Edition, MicroStrategy Incorporated, 2002.

11. Oracle9i Application Developer's Guide - Fundamentals, Release 2 (9.2), Oracle Corporation, 2002.
12. Oracle9*i* Security Overview Release 2 (9.2). Oracle Corporation 2002.

Recenzent: Prof. dr hab. inż. Leszek Borzemski

Wpłynęło do Redakcji 9 czerwca 2003 r.

Abstract

The research describes the process of implementation of a security system that controls access to the detail information in a ROLAP data warehouse. The process of creating the security policy is often led by requirements of two kinds: law regulations and needs specific for a given company.

The *Human Resources Analysis System* is presented and then improved to provide appropriate access control procedures. Fig.1 shows fragment of the data warehouse schema which is the subject in the securing process. Four stages of the system development are discussed:

1. Authorization by organizational structure.
2. Secure individual access to private data (security tables and the security view for an organizational hierarchy are shown in fig. 2).
3. Authorization by employee dimension.
4. Secure access to personal data.

There are many database and application mechanisms that can be used to implement the efficient, flexible and easy to maintain secure data warehouse system [3]. This research presents two ways of implementing the required security policies: secured database views and virtual private databases. The summary states the advantages and disadvantages of each solution.

Adresy

Marcin GORAWSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, M.Gorawski@zti.iinf.polsl.gliwice.pl .

Jacek FRĄCZEK: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, jacekf@zeus.polsl.gliwice.pl .