STUDIA INFORMATICA 2001
Volume 22 Number 1 (43)

Stanisław KOZIELSKI
Politechnika Śląska, Instytut Informatyki

MODELE I MECHANIZMY BEZPIECZEŃSTWA W SYSTEMACH BAZ DANYCH

Streszczenie. W pracy omówiono dwa podstawowe modele bezpieczeństwa stosowane w systemach baz danych: model uznaniowy (dyskrecjonalny) oraz model obowiązkowy (mandatoryjny).W opisie modelu uznaniowego zwrócono uwagę na definiowanie reguł dostępu, określających uprawnienia podmiotów do dostępu do obiektów bazy danych. Przedstawiono przykłady implementacji mechanizmów modelu uznaniowego w komercyjnych systemach zarządzania bazami danych. W opisie modelu obowiązkowego przedstawiono hierarchię klas dostępności przypisywanych podmiotom i obiektom bazy danych oraz reguły sterowania dostępem określające warunki odczytu i zapisu danych w powiązaniu z klasami dostępności. Szczególną uwagę zwrócono na wielopoziomowe systemy bezpieczne i konieczność rozszerzenia reguł modelu relacyjnego dla poprawnej implementacji takich systemów.

SECURITY MODELS AND MECHANISMS IN DATABASE SYSTEMS

Summary. Two security models applied in the database systems are presented in the paper: discretionary model and mandatory model. Discretionary access control is the basic mechanism applied in the most DBMS. In the description of mandatory model special attention is paid to multilevel secure relational data model.

1. Wprowadzenie

Bazy danych są sercem większości systemów informatycznych. Przechowywane w nich dane mogą być narażone na wiele rodzajów zagrożeń. Do najważniejszych zaliczamy:

- nielegalny odczyt danych przez nieuprawnionych użytkowników,
- niepoprawne operacje modyfikacji danych, które mogą być skutkiem:
  - umyślnego działania nieuprawnionych użytkowników,

- przypadkowych omyłek użytkowników,
- braku właściwej kontroli przy współbieżnym dostępie do danych wielu użytkowników,
- błędów oprogramowania lub awarii systemów komputerowych,

• zniszczenie danych w przypadku poważnych awarii sprzętu komputerowego.

Bezpieczeństwo baz danych rozpatruje się zazwyczaj w następujących obszarach:

1. Identyfikacja i autentyfikacja użytkowników
   Każdy użytkownik bazy danych musi zostać na wstępie zidentyfikowany w systemie komputerowym. Autentyfikacja jest procesem weryfikacji tożsamości użytkownika w trakcie wchodzenia (logowania) do systemu.
2. Kontrola dostępu
   Realizowana jest według określonej polityki zapewnienia bezpieczeństwa. W niniejszej pracy zostaną omówione dwa modele kontroli dostępu.
3. Ochrona integralności danych
   Mechanizmy zapewniające integralność baz danych można podzielić na mechanizmy integralności semantycznej oraz mechanizmy integralności transakcyjnej [3, 5, 7].
   Semantyczne więzy integralności definiują poprawny stan baz danych pomiędzy kolejnymi operacjami na bazie, stąd też umożliwiają ochronę, w pewnym zakresie, przed niepoprawną (umyślną lub przypadkową) modyfikacją danych. Mechanizmy integralności transakcyjnej chronią spójność bazy danych w warunkach współbieżnie realizowanych operacji na bazie przez wielu użytkowników, a także w przypadku błędów oprogramowania i awarii sprzętowych.
4. Monitorowanie operacji na bazie danych
   Wszystkie działania użytkowników istotne z punktu widzenia bezpieczeństwa systemu winny być monitorowane i rejestrowane. Analiza wyników takiej rejestracji może służyć do oceny poprawności przyjętej polityki bezpieczeństwa.

Celem niniejszej pracy jest przedstawienie i porównanie modeli i mechanizmów bezpieczeństwa w zakresie kontroli dostępu do danych. Znane i stosowane są dwa podstawowe modele: model dyskrecjonalny (*discretionary security*), w którym jest realizowana uznaniowa kontrola dostępu oraz model mandatoryjny (*mandatory security*) z obowiązkową kontrolą dostępu.

W opisie obu modeli stosowane są określenia *obiekt oraz podmiot* [2, 6]. Obiektem w kontekście bezpieczeństwa będziemy nazywali pasywną jednostkę, która zawiera lub jest w stanie przyjąć informację. Może to być cała baza danych, relacja, krotka lub wartość

atrybutu (tablica, perspektywa, wiersz, kolumna) lub ogólnie fakt reprezentowany w bazie danych. W ramach polityki bezpieczeństwa obiekt jest przedmiotem ochrony. Podmiotem jest użytkownik (osoba lub proces działający w imieniu użytkownika). Podmioty są odpowiedzialne za zmiany stanu bazy danych oraz przepływy informacji między różnymi obiektami i podmiotami.

2. Dyskrecjonalny model bezpieczeństwa

2.1. Ogólna charakterystyka uznaniowej kontroli dostępu

W modelu dyskrecjonalnym zakłada się, że dla każdego podmiotu zdefiniowano reguły dostępu określające uprawnienia podmiotu do obiektów systemu.

Przyjmijmy następujące oznaczenia:

$O = \{o_1, o_2, ..., o_k\}$ - zbiór obiektów (np. tablic bazy danych),

$S = \{s_1, s_2, ..., s_l\}$ - zbiór podmiotów (użytkowników bazy danych),

$T = \{t_1, t_2, ..., t_m\}$ - zbiór rodzajów dostępu (operacje takie jak select, insert, update, i in.),

$P = \{p_1, p_2, ..., p_n\}$ - zbiór predykatów, z których każdy definiuje zakres dostępu określonego podmiotu do określonego obiektu (zbiór P jest opcjonalny).

Regułami dostępu nazywamy krotki o postaci <o,s,t,p>.

Zdefiniujemy ponadto funkcję f

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

w celu określenia czy uprawnienie określone w krotce <o,s,t,p> jest ważne, czy nie.

Dla pewnej krotki <o,s,t,p>, jeśli wartością funkcji f(o,s,t,p) jest True, to podmiot s ma uprawnienie t do dostępu do obiektu o w zakresie zdefiniowanym przez predykat p.

Reguły dostępu są zazwyczaj przechowywane w postaci macierzy, w której w najprostszym przypadku wiersze opisują podmioty, kolumny opisują obiekty, zaś w przecięciu wiersza i kolumny umieszczona jest reguła dostępu, określająca uprawnienie podmiotu do obiektu.

Mechanizmy modeli dyskrecjonalnych wprowadzono w większości komercyjnych SZBD. W implementacji tych mechanizmów stosowane jest zazwyczaj jedno z dwu podejść:

- modyfikacja pytania użytkownika przez dodanie do niego stosownych warunków bezpieczeństwa (rozwiązanie to zastosowano po raz pierwszy w systemie Ingres),
- oddzielenie użytkownika od podstawowych tablic (relacji) przez zestaw perspektyw, w definicjach których wprowadza się ograniczenia związane z bezpieczeństwem (rozwiązanie zastosowane w Systemie R [5] i systemach na nim wzorowanych).

Ważną własnością modeli dyskrecjonalnych jest możliwość nadawania przez określonego użytkownika posiadanych przez niego praw (lub ich części) innym użytkownikom. Takie nadawanie może być następnie kaskadowo kontynuowane. Prawa te mogą być również odbierane.

Kaskadowe nadawanie i odbieranie uprawnień stwarza przy tym zagrożenie niejednoznaczności, np. w sytuacji, gdy dany użytkownik otrzymał pewne uprawnienie kilkoma różnymi drogami

2.2. Implementacja uznaniowych mechanizmów kontroli dostępu do danych

Stosowane w bazach danych uznaniowe mechanizmy kontroli dostępu do danych są w głównej mierze implementowane w serwerach baz danych. Pewien zakres zabezpieczeń jest też realizowany w aplikacjach bazodanowych.

2.2.1. Mechanizmy kontroli dostępu na poziomie serwera bazy danych

Mechanizmy kontroli dostępu zwykle obejmują również identyfikację i autentyfikację użytkowników. Pierwszym krokiem w takich działaniach jest utworzenie identyfikatora (konta) użytkownika przez administratora systemu. Operacji tej towarzyszy określenie tajnego hasła użytkownika. Istniejącym użytkownikom przyznawane są uprawnienia, które są zwykle dzielone na dwie kategorie:

1) uprawnienie o charakterze ogólnym odnoszące się do całej bazy danych (wszystkich jej obiektów),
2) uprawnienia szczegółowe, określające prawo do wykonywania określonej operacji na określonym obiekcie bazy danych.

Ad 1) Tworzenie kont użytkowników i przyznawanie im uprawnień ogólnych może przebiegać nieco inaczej w różnych Systemach Zarządzania Bazami Danych (SZBD). Przykładowo w systemie Oracle konto użytkownika jest tworzone poleceniem (w najprostszej postaci):

CREATE USER <użytkownik> INDENTIFIED BY <hasło>,

zaś uprawnienia ogólne (tu nazywane systemowymi) są przyznawane za pomocą polecenia (w najprostszej postaci):

GRANT <uprawnienie systemowe> TO <użytkownik>.

Przykłady uprawnień systemowych:

CREATE SESSION	- prawo do nawiązywania połączeń z bazą danych,
CREATE TABLE	- prawo do tworzenia tablic,
UPDATE ANY TABLE	- prawo do aktualizacji tablic,
SELECT ANY TABLE	- prawo do wyszukiwania danych z tablic.

W systemie Informix operacja tworzenia użytkowników jest powiązana z nadaniem im ogólnych uprawnień (nazywanych tu klasą użytkowników):

GRANT <klasa użytkownika> TO <użytkownik> [IDENTIFIED BY <hasło>],

gdzie klasy użytkowników określają między innymi:

CONNECT - prawo dostępu do tablic,
RESOURCE - prawo tworzenia tablic i udostępniania ich innym użytkownikom,
DBA - wszelkie prawa w bazie danych.

Ad 2) Jak wspomniano wyżej, uprawnienia szczegółowe dotyczą określonych operacji na określonych obiektach bazy danych, stąd uprawnienia te nazywane są uprawnieniami obiektowymi. Są one przyznawane poleceniem w języku SQL o postaci:

GRANT <uprawnienie obiektowe> ON <obiekt> TO <użytkownik> [WITH GRANT OPTION].

W dominujących obecnie serwerach baz danych opartych na języku SQL do operacji będących przedmiotem uprawnień należy większość instrukcji tego języka, w tym na przykład: SELECT, INSERT, DELETE, INDEX, ALTER, UPDATE [<lista kolumn>].

Uprawnienia mogą być również odbierane; umożliwia to instrukcja REVOKE.

Specyficznym aspektem rozważanych zagadnień jest umożliwienie propagacji uprawnień, tzn. przekazanie użytkownikowi prawa do przekazania nadanych mu uprawnień innym użytkownikom (opcja WITH GRANT OPTION). Wykorzystanie tej możliwości może w sposób mało kontrolowany rozszerzać grono użytkowników operujących na pewnych zasobach baz danych. Z drugiej strony natomiast odwołanie przyznanych pewnemu użytkownikowi uprawnień z taką opcją może wywołać kaskadowy efekt odbierania uprawnień innym użytkownikom, którzy dzięki tej opcji je otrzymali.

2.2.2. *Perspektywy*

Do szczególnych obiektów, objętych nadawaniem uprawnień, należą perspektywy (VIEWS). Mechanizm perspektyw pozwala tworzyć wirtualne tablice udostępniające użytkownikowi fragmenty zasobów jednej lub wielu tablic rzeczywistych w formie prostej lub przetworzonej. Przy wykorzystaniu tego mechanizmu mogą być przyznawane pewnym użytkownikom prawa dostępu nie do tablic rzeczywistych, a tylko do perspektyw. Pozwala to na wprowadzenie bardzo różnorodnych ograniczeń, w tym np. udostępnienie tylko wybranych kolumn bądź też wybranych wierszy tablic.

2.2.3. *Limity zasobów – profile użytkowników*

Poza ograniczeniami dostępu do danych (wynikającymi z przyznanych uprawnień) na użytkowników mogą być nakładane ograniczenia dotyczące zasobów systemowych, kontrolowanych przez System Zarządzania Bazą Danych. Należą do nich np.:

- ilość czasu procesora przeznaczonego na obsługę zadań określonego użytkownika,

- liczba równoczesnych sesji otwartych przez użytkownika,
- liczba odczytów (logicznych) z dysku przez użytkownika,
- dopuszczalny czas bez wykonywania operacji na bazie danych.

Zestaw nałożonych ograniczeń tworzy tzw. profil użytkownika.

2.2.4. Mechanizmy kontroli dostępu na poziomie aplikacji klienta

Przedstawione dotychczas mechanizmy nadawania uprawnień i kontroli za ich pomocą dostępu do bazy danych są realizowane w serwerach baz danych. Ochrona systemów bazodanowych może być również realizowana w aplikacjach klienta, czyli programach użytkowych wykonywanych w stacjach roboczych, korzystających z bazy danych za pośrednictwem serwera.

Mechanizmy ochrony są w takim przypadku tworzone przez projektanta aplikacji. Projektant dokonuje zazwyczaj kategoryzacji użytkowników aplikacji i poszczególnym kategoriom zapewnia dostęp do określonych funkcji aplikacji. Dostępność funkcji jest realizowana przez odpowiednie ukształtowanie menu programu. Funkcje niedostępne dla danego użytkownika są na ogół niewidoczne, czasami program nie pozwala ich wybrać ("wygaszone" pozycje w menu).

Użytkownicy aplikacji powinni być równocześnie zdefiniowani w systemie jako użytkownicy bazy danych, z której korzysta aplikacja. Bardzo ważne jest w takim przypadku precyzyjne skorelowanie uprawnień definiowanych na poziomie aplikacji z uprawnieniami definiowanymi na poziomie serwera bazy danych.

2.2.5. Monitorowanie bazy danych

Z uznaniową kontrolą dostępu powiązana jest dostępna w niektórych serwerach baz danych możliwość tzw. audytu, czyli obserwacji i rejestrowania informacji o działaniach użytkowników. Taka funkcja, którą określimy jako monitorowanie, może np. obejmować:

- monitorowanie operacji: śledzenie wskazanych operacji (instrukcji) języka SQL (wykonanych lub odrzuconych),
- monitorowanie uprawnień: śledzenie wykorzystania uprawnień systemowych przez wskazanych użytkowników,
- monitorowanie obiektów: śledzenie wykonania wskazanych operacji (instrukcji) SQL na wskazanych obiektach.

Monitorowanie może być wykorzystane do podniesienia bezpieczeństwa systemu, przede wszystkim poprzez kontrolę działań użytkowników, którzy próbują przekraczać przyznane im uprawnienia. Poza tym monitorowanie jest wykorzystywane do strojenia serwera bazy danych.

3. Mandatoryjny model bezpieczeństwa

Model mandatoryjny wymusza obowiązkową kontrolę dostępu do danych. Kontrola taka wymaga dokonania klasyfikacji obiektów i podmiotów. Każdemu obiektowi przypisuje się pewien poziom tajności, który będziemy też nazywać klasą dostępności lub krótko klasą [2, 3, 6]. Przykładem klasyfikacji obiektów może być następująca hierarchia poziomów tajności (klas):

- ściśle tajne = 4
- tajne = 3
- poufne = 2
- niesklasyfikowane = 1.

Podobnie każdemu podmiotowi jest przypisany pewien poziom uprawnień dostępu do określonych klas danych, inaczej przepustka (*clearance*). Charakteryzuje on poziom zaufania przypisany temu podmiotowi, tzn. stopień pewności, że podmiot ten nie ujawni innym wrażliwych danych. W dalszych przykładach przyjmiemy, że przykładowe poziomy uprawnień (przepustki) odpowiadają przedstawionym wyżej poziomom tajności.

Sterowanie dostępem w oparciu o model mandatoryjny bazuje na dwu regułach [3]:

1. Podmiot s może odczytać obiekt o, jeśli poziom uprawnień (przepustka) podmiotu nie jest mniejszy od poziomu tajności (klasy) obiektu, tzn. przepustka(s) ≥ klasa(o).

2. Podmiot s może zapisać lub modyfikować obiekt o, jeśli poziom uprawnień podmiotu jest równy poziomowi tajności obiektu, tzn.: przepustka(s) = klasa(o).

Interpretacja reguły 1 jest intuicyjnie oczywista, natomiast regułę 2 można objaśnić następująco: wszystkie dane zapisane przez podmiot s otrzymują poziom tajności równy poziomowi uprawnień tego podmiotu. W ten sposób zabezpieczamy się przed sytuacją, w której nieuczciwy użytkownik o wysokich uprawnieniach skopiuje tajne dane nadając im niższy poziom tajności i udostępniając je w ten sposób użytkownikom o niższych uprawnieniach.

Reguła 2 jest przedstawiana często [1, 2, 6] w ogólniejszej postaci:

2'. Podmiot s może zapisać lub modyfikować obiekt o, jeśli poziom uprawnień podmiotu nie jest większy od poziomu tajności obiektu, tzn.: przepustka(s) ≤ klasa(o).

Ograniczenia dostępu wynikające z powyższych reguł powodują, że użytkownicy o różnych poziomach uprawnień widzą tę samą bazę danych w różny sposób, tzn. jako wynik identycznie sformułowanych zapytań otrzymują różne obrazy tej samej bazy danych. Operowanie na takiej bazie danych wymaga wprowadzenia dodatkowych zasad, określających tzw. wielopoziomowy bezpieczny relacyjny model danych [2, 3, 6] (*Multilevel Secure Relational Data Model*).

3.1. Wielopoziomowy bezpieczny relacyjny model danych

Model wielopoziomowy przedstawimy najpierw na podstawie wybranych przykładów. Załóżmy, że koncepcję obowiązkowej kontroli dostępu zastosujemy do relacji Projekty, w której przechowywane będą dane o identyfikatorach projektów (Id_projektu), ich nazwach, nazwiskach kierowników i funduszach. Przyjmijmy wstępnie, że obiektami, do których dostęp będzie kontrolowany, będą pojedyncze krotki. Każdej krotce przypiszemy określony poziom tajności (klasę) zgodnie z przyjętą uprzednio hierarchią (ściśle tajne =4, tajne =3, poufne =2, niesklasyfikowane =1).

Przykładową zawartość rozważanej relacji ilustruje rys.1.

Projekty

Id_projektu	Nazwa	Kierownik	Fundusze	Klasa
P1	Zasilacz	Grabski	12 000	3
P2	Generator	Adamski	7 000	2
P3	Sterownik	Jaworek	20 000	3
P4	Reaktor	Borowy	35 000	4
P5	Regulator	Lipski	15 000	2

Rys. 1. Przykładowa zawartość relacji Projekty
Fig. 1. Example of multilevel relation Projects

Załóżmy teraz, że dwaj użytkownicy U1 i U2 mają poziomy uprawnień (przepustkę) odpowiednio 3 i 2 (tzn. prawo dostępu do danych tajnych oraz poufnych). Różnica poziomów uprawnień spowoduje, że zapytanie o wszystkie projekty

SELECT * FROM Projekty

zadane przez użytkownika U1 da w efekcie relację przedstawioną na rys. 2, zaś to samo zapytanie zadane przez użytkownika U2 da w wyniku relację przedstawioną na rys. 3. Użytkownicy o różnych poziomach uprawnień widzą więc tę samą relację odmiennie, stosownie do swoich przepustek.

Id_projektu	Nazwa	Kierownik	Fundusze	Klasa
P1	Zasilacz	Grabski	12 000	3
P2	Generator	Adamski	7 000	2
P3	Sterownik	Jaworek	20 000	3
P5	Regulator	Lipski	15 000	2

Rys. 2. Wynik zapytania zadanego przez użytkownika z poziomem uprawnień równym 3
Fig. 2. Result of query put by user with clearance equal 3

Id_projektu	Nazwa	Kierownik	Fundusze	Klasa
P2	Generator	Adamski	7 000	2
P5	Regulator	Lipski	15 000	2

Rys. 3. Wynik zapytania zadanego przez użytkownika z poziomem uprawnień równym 2
Fig. 3. Result of query put by user with clearance equal 2

Przedstawiona koncepcja wymaga wprowadzenia nowej terminologii [2, 3, 6]: relacja Projekty jest przykładem **relacji wielopoziomowej** (*multi-level*), zaś fakt, że te same dane (zawartość relacji Projekty) wyglądają różnie dla różnych użytkowników, nazywany jest **wieloinstancyjnością** (*polyinstantiation*). Rysunki 2 i 3 przedstawiają więc dwie różne instancje relacji Projekty, odpowiadające poziomom uprawnień (przepustkom) równym odpowiednio 3 i 2.

Rozważmy następnie bardziej szczegółowy poziom kontroli dostępu do bazy danych. Przyjmijmy, że obiektami o obowiązkowo kontrolowanym dostępie będą pojedyncze dane, tzn. wartości atrybutów. W takim przypadku każdemu atrybutowi musi towarzyszyć klasa (tzn. poziom tajności wartości tego atrybutu). Rozpatrywaną uprzednio relację Projekty możemy teraz przedstawić jak na rys. 4.

Projekty

Id_projektu	K1	Nazwa	K2	Kierownik	K3	Fundusze	K4	Klasa
P1	2	Zasilacz	3	Grabski	3	12 000	3	3
P2	2	Generator	2	Adamski	2	7 000	2	2
P3	3	Sterownik	3	Jaworek	3	20 000	4	4
P4	4	Reaktor	4	Borowy	4	35 000	4	4
P5	2	Regulator	2	Lipski	2	15 000	3	2

Rys. 4. Przykładowa zawartość relacji Projekty w przypadku kontroli dostępu na poziomie pojedynczych danych
Fig. 4. Example of multilevel relation Projects with access control on individual attribute values level

Rezultaty identycznego jak uprzednio zapytania o wszystkie informacje o projektach

SELECT * FROM Projekty

zadanego przez użytkowników U1 i U2 o przepustkach odpowiednio 3 i 2 przedstawiono na rys. 5.

a) instancja dla poziomu uprawnień = 3

Id_projektu	K1	Nazwa	K2	Kierownik	K3	Fundusze	K4	Klasa
P1	2	Zasilacz	3	Grabski	3	12 000	3	3
P2	2	Generator	2	Adamski	2	7 000	2	2
P3	3	Sterownik	3	Jaworek	3	-	3	3
P5	2	Regulator	2	Lipski	2	15 000	3	2

b) instancja dla poziomu uprawnień = 2

Id_projektu	K1	Nazwa	K2	Kierownik	K3	Fundusze	K4	Klasa
P1	2	-	2	-	2	-	2	2
P2	2	Generator	2	Adamski	2	7 000	2	2
P5	2	Regulator	2	Lipski	2	-	2	2

Rys. 5. Dwie instancje wielopoziomowej relacji Projekty utworzone dla użytkowników U1 i U2 o różnych poziomach uprawnień

Fig. 5. Two instances of multilevel relation Projects created for users with different clearances

Porównując instancje relacji Projekty przedstawione na rys. 5, rys. 2 i rys. 3 można zauważyć, że:

- Krotki P3 w instancji a oraz P5 w instancji b zawierają pustą wartość atrybutu *Fundusze*, bowiem klasa tego atrybutu w rozważanych krotkach była wyższa od przepustki użytkowników, dla których utworzono odpowiednie instancje.
- W krotce P1 instancji b ujawniony został tylko numer projektu, bowiem klasa pozostałych atrybutów jest wyższa od przepustki użytkownika U2, dla którego utworzono tę instancję.
- Wszystkie nieujawnione wartości atrybutów (przyjmujące wartość null) otrzymują klasę równą poziomowi uprawnień (przepustce) użytkownika, dla którego utworzono daną instancję – konieczność spełnienia tego wymogu wynika z przedstawionego dalej formalnego opisu modelu wielopoziomowego.

Rozważmy następnie na trzech przykładach problem wprowadzania nowych krotek do wielopoziomowej relacji Projekty.

Przykład 1. Rozpatrzmy wykonanie instrukcji

INSERT INTO Projekty VALUES ('P6', 'Stabilizator', 'Orzeszek', 18000).

Zauważmy, że zgodnie z regułą 2 wprowadzenie tej krotki do relacji wymaga nadania jej klasy równej przepustce użytkownika wykonującego powyższą operację. Stąd też efekt wykonania tej operacji będzie różnie widoczny w instancjach relacji Projekty (rys. 6 i 7), zależnie od tego, który z użytkowników U1, U2 wykonał tę operację:

a) jeśli instrukcję INSERT wykonał użytkownik U1 (z przepustką równą 3), efekt wprowadzenia krotki będzie widoczny tylko w instancji odpowiadającej poziomowi uprawnień równemu 3,

b) jeśli instrukcję INSERT wykonał użytkownik U2 (z przepustką równą 2), efekt wprowadzenia krotki będzie widoczny w obu instancjach tej relacji.

a) instancja dla poziomu uprawnień = 3

Id_projektu	K1	Nazwa	K2	Kierownik	K3	Fundusze	K4	Klasa
P1	2	Zasilacz	3	Grabski	3	12 000	3	3
P2	2	Generator	2	Adamski	2	7 000	2	2
P3	3	Sterownik	3	Jaworek	3	-	3	3
P5	2	Regulator	2	Lipski	2	15 000	3	2
P6	3	Stabilizator	3	Orzeszek	3	18 000	3	3

b) instancja dla poziomu uprawnień = 2

Id_projektu	K1	Nazwa	K2	Kierownik	K3	Fundusze	K4	Klasa
P1	2	-	2	-	2	-	2	2
P2	2	Generator	2	Adamski	2	7 000	2	2
P5	2	Regulator	2	Lipski	2	-	2	2

Rys.6. Dwie instancje relacji Projekty w przypadku wprowadzenia projektu P6 przez użytkownika z poziomem uprawnień równym 3

Fig. 6. Two instances of multilevel relation Projects in case of insertion project P6 by user with clearance equal 3

a) instancja dla poziomu uprawnień = 3

Id_projektu	K1	Nazwa	K2	Kierownik	K3	Fundusze	K4	Klasa
P1	2	Zasilacz	3	Grabski	3	12 000	3	3
P2	2	Generator	2	Adamski	2	7 000	2	2
P3	3	Sterownik	3	Jaworek	3	-	3	3
P5	2	Regulator	2	Lipski	2	15 000	3	2
P6	2	Stabilizator	2	Orzeszek	2	18 000	2	2

b) instancja dla poziomu uprawnień = 2

Id_projektu	K1	Nazwa	K2	Kierownik	K3	Fundusze	K4	Klasa
P1	2	-	2	-	2	-	2	2
P2	2	Generator	2	Adamski	2	7 000	2	2
P5	2	Regulator	2	Lipski	2	-	2	2
P6	2	Stabilizator	2	Orzeszek	2	18 000	2	2

Rys.7. Dwie instancje relacji Projekty w przypadku wprowadzenia projektu P6 przez użytkownika z poziomem uprawnień równym 2

Fig. 7. Two instances of multilevel relation Projects in case of insertion project P6 by user with clearance equal 2

Przykład 2. Przyjmując jako wyjściową postać relacji Projekty z rys. 4 i jej dwie instancje z rys. 5 rozpatrzmy wykonanie instrukcji

INSERT INTO Projekty VALUES ('P2', 'Generator', 'Sosnowski', 7000)

przez użytkownika U1 (z przepustką równą 3).

Rozważać można dwa warianty reakcji systemu interpretującego tę instrukcję:

- system informuje użytkownika U1 o istnieniu krotki z identycznym kluczem głównym (P2) i odrzuca instrukcję INSERT,
- system wprowadza krotkę – postać dwu instancji relacji Projekty po takiej operacji przedstawia rys. 8.

a) instancja dla poziomu uprawnień = 3

Id_projektu	K1	Nazwa	K2	Kierownik	K3	Fundusze	K4	Klasa
P1	2	Zasilacz	3	Grabski	3	12 000	3	3
P2	2	Generator	2	Adamski	2	7 000	2	2
P2	3	Generator	3	Sosnowski	3	7 000	3	3
P3	3	Sterownik	3	Jaworek	3	-	3	3
P5	2	Regulator	2	Lipski	2	15 000	3	2

b) instancja dla poziomu uprawnień = 2

Id_projektu	K1	Nazwa	K2	Kierownik	K3	Fundusze	K4	Klasa
P1	2	-	2	-	2	-	2	2
P2	2	Generator	2	Adamski	2	7 000	2	2
P5	2	Regulator	2	Lipski	2	-	2	2

Rys.8. Dwie instancje relacji Projekty po wprowadzeniu nowego projektu P2 przez użytkownika z poziomem uprawnień równym 3

Fig. 8. Two instances of multilevel relation Projects in case of insertion new project P2 by user with clearance equal 3

Zauważmy, że wprowadzenie do instancji a drugiej krotki z identyfikatorem projektu P2 narusza więzy integralności relacji Projekty (powtórzenie wartości klucza głównego). Będzie to poprawne, jeśli rozszerzymy definicję klucza głównego tej relacji dodając do identyfikatora projektu jego klasę K1. Zauważmy też, że wprowadzenie nowej krotki jest niewidoczne dla użytkownika o niższym poziomie uprawnień.

Przykład 3. Przyjmując jako wyjściową postać relacji Projekty z rys. 4 i jej dwie instancje z rys. 5 rozpatrzmy wykonanie instrukcji

INSERT INTO Projekty VALUES ('P3', 'Prostownik', 'Bukowy', 22000)

przez użytkownika U2 (z przepustką równą 2).

Zauważmy, że podobnie jak w przykładzie poprzednim krotka z kluczem głównym P3 już istnieje. Jednakże w tym przypadku system interpretujący instrukcję INSERT nie może poinformować użytkownika U2 o istnieniu takiej krotki, bowiem jej klasa jest równa 4, a więc musi być ona niewidoczna dla użytkownika U2 o poziomie uprawnień równym 2. Niemożliwe jest więc w tym przypadku odrzucenie instrukcji INSERT, nastąpiłoby bowiem ujawnienie informacji o istnieniu krotki z takim kluczem. Postać obu rozważanych instancji relacji Projekty po wprowadzeniu tej krotki przedstawia rys. 9.

a) instancja dla poziomu uprawnień = 3

Id_projektu	K1	Nazwa	K2	Kierownik	K3	Fundusze	K4	Klasa
P1	2	Zasilacz	3	Grabski	3	12 000	3	3
P2	2	Generator	2	Adamski	2	7 000	2	2
P3	2	Prostownik	2	Bukowy	2	22 000	2	2
P3	3	Sterownik	3	Jaworek	3	-	3	3
P5	2	Regulator	2	Lipski	2	15 000	3	2

b) instancja dla poziomu uprawnień = 2

Id_projektu	K1	Nazwa	K2	Kierownik	K3	Fundusze	K4	Klasa
P1	2	-	2	-	2	-	2	2
P2	2	Generator	2	Adamski	2	7 000	2	2
P3	2	Prostownik	2	Bukowy	2	22 000	2	2
P5	2	Regulator	2	Lipski	2	-	2	2

Rys.9. Dwie instancje relacji Projekty po wprowadzeniu nowego projektu P3 przez użytkownika z poziomem uprawnień równym 2

Fig. 9. Two instances of multilevel relation Projects in case of insertion new project P3 by user with clearance equal 2

Zauważmy, że wprowadzenie rozważanej krotki P3 narusza, podobnie jak w przykładzie poprzednim, warunki integralności relacji Projekty (naruszenie unikalności klucza głównego).

Przedstawione przykłady pokazują, że wprowadzenie modelu wielopoziomowego i dopuszczenie wieloinstancyjności powoduje naruszenie niektórych podstawowych własności modelu relacyjnego. Stwarza to konieczność rozszerzenia definicji modelu relacyjnego w zastosowaniu do opisu modelu wielopoziomowego [1, 2, 6].

3.2. Formalna definicja modelu wielopoziomowego

Schemat relacji dla modelu wielopoziomowego jest definiowany [2, 6] następująco:

$R(A_1, K_1, ..., A_n, K_n, KK)$,

gdzie każde A_i jest atrybutem, każde K_i jest klasą dostępności atrybutu A_i, zaś KK jest klasą całych krotek. Atrybuty A_i zdefiniowane są nad dziedzinami $dom(A_i)$ jak w klasycznym modelu relacyjnym. Dziedzinę klas dostępności K_i oraz klasy krotek KK tworzy zbiór $\aleph=\{k\}$ przyjętych w danej instytucji wartości klas (poziomów tajności).

W rozpatrywanym modelu wielopoziomowym jednemu schematowi relacji R odpowiada zestaw instancji relacji r, po jednej dla każdej klasy $k \in \aleph$. Instancja relacji r_k składa się ze zbioru krotek postaci: $(a_1, k_1, .. a_n, k_n, kk)$, gdzie każde $a_i \in dom(A_i)$, $k_i \leq k$, $k_i \in \aleph$, kk jest kresem górnym zbioru $\{k_i, i=1..n\}$.

Przedstawiona definicja instancji relacji określa istotę różnicy modelu wielopoziomowego i zwykłego modelu relacyjnego.

Na podstawie powyższej definicji można określić sposób odczytu krotki t przez podmiot s: każda wartość $t[A_i]$ jest widoczna dla podmiotu s, jeśli przepustka(s) $\geq t[K_i]$, w przeciwnym przypadku $t[A_i]$ jest zastępowane przez wartość pustą, zaś $t[K_i]$ przez wartość k.

Własności wielopoziomowego modelu bezpiecznego określone są dodatkowo przez więzy spójności: spójność encji oraz spójność wartości pustych. Można je wyprowadzić ze znanych dla klasycznego modelu relacyjnego pojęć klucza i więzów referencyjnych.

W modelu relacyjnym wartość klucza jednoznacznie identyfikuje każdą krotkę, zaś atrybuty niekluczowe są funkcyjnie zależne od klucza. W modelu wielopoziomowym taki klucz jest nazywany kluczem jawnym (*apparent key*).

Załóżmy, że $R(A_1, K_1, ...A_n, K_n, KK)$ oznacza schemat relacji w modelu wielopoziomowym, zaś A jest zbiorem atrybutów tworzących klucz jawny ($A \subseteq \{A_1, A_2, ... A_n,\}$).

Własność spójności encji. Relacja r modelu wielopoziomowego spełnia własność spójności encji wtedy i tylko wtedy, jeśli dla wszystkich instancji r_k oraz $t \in r_k$ zachodzi:

1. $A_i \in A \Rightarrow t[A_i] \neq null$
2. $A_i, A_j \in A \Rightarrow t[K_i] = t[K_j]$
3. $A_i \notin A \Rightarrow t[K_i] \geq t[K_A]$ (K_A jest klasą dostępności klucza A).

Własność spójności encji stanowi, że klucz jawny nie może mieć wartości pustych, wszystkie jego składowe muszą mieć jednakową klasę dostępności, a klasa ta nie może przewyższać klas innych atrybutów.

Własność spójności wartości pustych. Relacja r spełnia własność spójności wartości pustych wtedy i tylko wtedy, jeśli dla każdej instancji r_k relacji r spełnione są następujące warunki:

1. Dla każdej krotki $t \in r_k$ zachodzi:
   $t[A_i] = null \Rightarrow t[K_i] = t[K_A]$.
2. Relacja r_k nie zawiera dwu różnych krotek takich, że jedna „podciąga" drugą. Krotka t podciąga krotkę s, jeśli dla każdego atrybutu A_i albo $t[A_i, K_i] = s[A_i, K_i]$, albo $t[A_i] \neq null$ i $s[A_i] = null$.

Spójność wartości pustych oznacza, że wartości puste muszą być sklasyfikowane w tej samej klasie dostępności co klucz oraz dla podmiotów o wyższych poziomach uprawnień, wartości puste widoczne przy niższych uprawnieniach są automatycznie zastępowane przez właściwe wartości (niepuste).

Przedstawione powyżej definicje określają tylko podstawowe własności modelu wielopoziomowego. W literaturze zdefiniowano szereg dodatkowych, szczegółowych cech tego modelu – przegląd takich prac zawarto np. w [2, 6].

3.3. Przykłady wielopoziomowych systemów bezpiecznych

Idee modelu mandatoryjnego zostały wykorzystane w budowie kilku eksperymentalnych systemów zarządzania bazami danych określanych jako bezpieczne (*secure*) lub wiarygodne (*trusted*). Najbardziej znany jest projekt SeaView, zrealizowany wspólnie przez Stanford Research Institute, Oracle oraz Gemini Computers [4, 6]. W projekcie tym zrealizowano koncepcję wielopoziomowego bezpiecznego relacyjnego systemu zarządzania bazą danych. Obowiązkowa kontrola dostępu jest realizowana na najniższym poziomie pojedynczych wartości atrybutów. Wielopoziomowe relacje istnieją przy tym tylko na poziomie logicznym i są dekomponowane do jednopoziomowych relacji bazowych. Relacje bazowe są przechowywane przy użyciu konwencjonalnego systemu zarządzania bazą danych (Oracle). Wielopoziomowe relacje są tworzone jako perspektywy (views), przy wykorzystaniu relacji jednopoziomowych. Operacje na zawartości relacji wielopoziomowych są kontrolowane według reguł modelu mandatoryjnego (obowiązkowa kontrola dostępu).

Oprócz obowiązkowej kontroli dostępu (do zasobów bazy danych) w projekcie SeaView zrealizowano dodatkowo uznaniową kontrolę dostępu (model dyskrecjonalny). Kontrola ta wykonywana jest na wyższym, wstępnym poziomie i ma na celu określenie użytkowników i ich grup, uprawnionych (bądź nieuprawnionych) do dostępu (w wyróżnionym trybie) do poszczególnych obiektów bazy danych.

Przykłady innych wielopoziomowych bezpiecznych relacyjnych systemów zarządzania bazami danych to [6]:

- Lock Data Views (LDV) opracowany w Honeywell Secure Computing Technology Center oraz MITRE,
- ASD_Views opracowany w TRW,
- Trusted Oracle 7 opracowany przez Oracle Corporation.

Pierwszy z nich realizuje obowiązkową kontrolę dostępu na poziomie całych krotek, natomiast drugi na poziomie zmaterializowanych perspektyw. Trusted Oracle 7 realizuje obowiązkową kontrolę dostępu przy współdziałaniu z systemem operacyjnym.

4. Podsumowanie

W pracy przedstawiono dwa różne podejścia do kontroli dostępu: model dyskrecjonalny z uznaniową kontrolą dostępu oraz model mandatoryjny z obowiązkową kontrolą dostępu.

Zaletą modelu dyskrecjonalnego jest jego elastyczność i łatwość dostosowania do specyfiki konkretnych systemów i aplikacji. Dlatego uznaniowa kontrola dostępu jest szeroko stosowana w dostępnych, komercyjnych systemach zarządzania bazami danych.

Wady modelu dyskrecjonalnego związane są z dwiema jego cechami:

- mechanizmy tego modelu pozwalają (w pewnym zakresie) na ustalanie warunków dostępu do danych przez samych użytkowników, tak więc odpowiedzialność za bezpieczeństwo systemu spoczywa po stronie użytkowników,
- model ten nie umożliwia ścisłej kontroli przepływu danych, co pozwala na obejście ograniczeń dostępu.

W modelu mandatoryjnym użytkownicy nie mają kontroli nad ustalaniem parametrów bezpieczeństwa. Podstawą tego modelu jest ścisła klasyfikacja użytkowników systemu i obiektów przechowywanych w bazie danych. Realizowana na tej podstawie kontrola dostępu zapewnia również kontrolę przepływu danych.

Wadą tego modelu jest natomiast sztywność jego reguł oraz możliwość zastosowania tylko w środowiskach, gdzie jest możliwa klasyfikacja (i odpowiednie oznakowanie) wszystkich pojawiających się danych.

W szeregu opracowywanych obecnie systemach podejmuje się próby łączenia polityk bezpieczeństwa stosowanych w obu przedstawionych modelach.

LITERATURA

1. Bell D. E., La Padula L. J.: Secure Computer Systems: Mathematical Foundations and Model. MITRE Technical Report M74-244, 1974.
2. Castano S., Fugini M. G., Martella G., Samarati P.: Database Security. Addison-Wesley Publishing Company, 1995.
3. Date C. J.: An Introduction to Database Systems (7th ed.). Addison-Wesley, 2000.
4. Denning D. E., Lunt T. F., Schell R. R., Shockley W. R., Heckman M.: The SeaView Security Model. Proc. 1988 Symp. On Research in Security and Privacy, IEEEComputer Society Press, 1988.
5. Elmasri R., Navathe S.: Fundamentals of Database Systems. Addison-Wesley, 2000.
6. Pernul G.: Database Security. In: Advances in Computers, Vol.38. Yovits M. C. (Ed.), Academic Press, 1994.
7. Silberschatz A., Korth H., Sudarshan S.: Database System Concepts. McGraw-Hill, 1996.

Recenzent: Prof. dr inż. Stefan Węgrzyn

Wpłynęło do Redakcji 7 maja 2001 r.

Abstract

Two security models applied in the database systems are presented in the paper: discretionary model and mandatory model. Discretionary access control is the basic mechanism applied in the most DBMS. The access rules presented in the paper include: user creation, privileges granting and revoking and view creation. These rules are formulated in SQL language. Mandatory model requires that objects and subjects in database system are assigned to certain security levels named classification (class) for objects and clearance (clear) for subjects. The relation between the clear of subject and the class of object determines if subject is allowed to read or write data item. For this reason the contents of database may appear different to users with different clearances. In the paper special attention is paid to multilevel secure relational data model. Different instances of relations which are results of selection or insertion of data from/into relation of database made by users (subjects) with different clearances are presented in many figures. The formal description of multilevel secure relational data model is also presented. Discretionary and mandatory models are compared in conclusions.