

Politechnika Wrocławska
Wydział Informatyki i Zarządzania
Instytut Sterowania i Techniki Systemów

Raport serii PRE 4/2001

**Model mechanizmu sterowania
i zarządzania dostępem
poprzez role
w systemach informatycznych.**

(rozprawa doktorska)

Grzegorz Kołaczek

Promotor: prof. dr hab. inż. Adam Grzech

Wrocław 2001

1. WSTĘP.....	4
1.1. WPROWADZENIE W TEMATYKĘ PRACY.....	4
1.2. CEL BADAWCZY.	6
1.3. PORZĄDEK PRACY.	9
1.4. ODNIESIENIA LITERATUROWE.	12
2. BEZPIECZEŃSTWO SYSTEMÓW INFORMATYCZNYCH.	16
2.1. ELEMENTY BEZPIECZEŃSTWA SYSTEMÓW INFORMATYCZNYCH.	16
2.2. STEROWANIE DOSTĘPEM.	20
2.3. ZARZĄDZANIE DOSTĘPEM.	23
2.4. PODSUMOWANIE.	23
3. PRZEGLĄD PODSTAWOWYCH POLITYK STEROWANIA DOSTĘPEM.....	24
3.1. UZNANIOWE STEROWANIE DOSTĘPEM - DAC.	25
3.1.1. Mechanizmy polityki DAC.	28
3.2. NARZUCONE STEROWANIE DOSTĘPEM - MAC.....	29
3.2.1. Mechanizmy MAC.....	31
3.3. PODSUMOWANIE.	32
4. POLITYKA I MECHANIZMY RBAC.	33
4.1. POSTULATY STEROWANIA DOSTĘPEM POPRZEZ ROLE.	33
4.2. WYBRANE MECHANIZMY POLITYKI RBAC.	36
4.2.1. Model RBAC'96 i ARBAC'97.....	36
4.2.2. Model mechanizmu dla ograniczeń w polityce RBAC.....	45
4.2.3. Model referencyjny mechanizmu RBAC.....	48
4.2.4. Model obiektowo zorientowanego RBAC.	52
4.3. PODSUMOWANIE.	54
5. MODEL STEROWANIA I ZARZĄDZANIA DOSTĘPEM OPARTY O POLITYKĘ RBAC.	56
5.1. LOGIKA FORMALNA.....	56
5.1.1. Normatywny aspekt sterowania dostępem.	57
5.2. DEFINICJA SYNTAKTYCZNA I SEMANTYCZNA JĘZYKA MODELU RMM.....	59
5.3. SEMANTYKA JĘZYKA <i>RMM</i> W ŚRODOWISKU RBAC.	62
5.4. ZASTOSOWANIE MODELU <i>RMM</i> DO OPISU RBAC.	67
5.4.1. Założenia wstępne.	67
5.5. PODSUMOWANIE.	77
6. MECHANIZM STEROWANIA DOSTĘPEM.	78
6.1. ANALIZA SPÓJNOŚCI ŻĄDAŃ PODMIOTÓW Z PRZYJĘTĄ POLITYKĄ.	78
6.2. PODSUMOWANIE.	83

7. MECHANIZMY ZARZĄDZANIA DOSTĘPEM.	84
7.1. SPÓJNOŚĆ WEWNĘTRZNA RÓL.	84
7.1.1. Zadanie rozszerzania zbioru akcji dozwolonych, zabronionych i obowiązkowych	84
7.1.2. Zadanie rozwiązania problemu sprzeczności norm	85
7.2. HIERARCHIA RÓL.....	89
7.3. OGRANICZENIA.	95
7.3.1. Ograniczenia na ilość aktywnych odwołań.	96
7.3.2. Role wzajemnie wykluczające się.....	97
7.3.3. Role wymagające siebie nawzajem.....	98
7.3.4. Role z jednoczesnym ograniczeniami na wzajemne wykluczanie się i wymaganie innej roli.....	99
7.4. MECHANIZMY DEFINIOWANIA POWIĄZAŃ ROLA-PRAWA DOSTĘPU I ROLA-PODMIOT.	99
7.4.1. Definiowanie i usuwanie podmiotów i obiektów.....	99
7.4.2. Definiowanie i usuwanie ról.	100
7.4.3. Asocjacja ról z podmiotami.....	102
7.5. PODSUMOWANIE.	104
8. PODSUMOWANIE.	105
DODATEK A. WYKAZ NAJWAŻNIEJSZYCH SKRÓTÓW WYSTĘPUJĄCYCH W PRACY.	110
DODATEK B. POMOCNICZE DEFINICJE Z ZAKRESU MATEMATYKI I LOGIKI.	111
DODATEK C. ELEMENTY OPRACOWANEGO MODELU STEROWANIA DOSTĘPEM.....	118
DODATEK D. BIBLIOGRAFIA.	123

1. Wstęp.

1.1. Wprowadzenie w tematykę pracy.

Rozwój systemów informatycznych (SI) i wzrost ich znaczenia dla funkcjonowania współczesnych organizacji powoduje, iż podstawowym zagadnieniem dla każdego systemu jest możliwość sterowania i koordynowania działań aktywnych w nim procesów. Jest tak, gdyż sposób w jaki realizowana jest koordynacja i sterowanie aktywnością procesów ma istotny wpływ na efektywność działania systemu jak i na jego bezpieczeństwo.

Bezpieczeństwo SI w ogólnym sensie polega na ochronie wartościowych zasobów systemu takich jak przetwarzane w nim informacje, dane oraz wchodzący w jego skład sprzęt i oprogramowanie. W tym kontekście zagadnienie sterowania działaniami procesów jest powiązane ze sterowaniem i zarządzaniem dostępem (ang. Access Control), które jest istotnym elementem bezpieczeństwa SI. Przez dostęp będziemy rozumieli możliwość wykonania operacji określonego typu na zasobach SI. Sterowanie dostępem zaś ma na celu zapobieżenie złośliwym i zamierzonym naruszeniom obowiązujących ograniczeń dostępu oraz zmniejszenie prawdopodobieństwa przypadkowych i mimowolnych nadużyć. Jest ono realizowane poprzez ograniczenie możliwości podmiotów do zbioru operacji niezbędnych do wykonywania zadań wyznaczonych przez politykę organizacji.

Skuteczność sterowania dostępem jest uzależniona od dwóch warunków wstępnych. Pierwszym z nich jest możliwość poprawnej identyfikacji aktywnych podmiotów. Tylko na podstawie sprawdzonej tożsamości, potwierdzonej w fazie inicjowania aktywności podmiotu w systemie, możliwe jest zagwarantowanie tego, iż podmioty będą uprawnione do korzystania wyłącznie z przysługujących im zasobów. Drugim warunkiem jest zapewnienie ochrony zasobom systemowym, które przechowują dane odnośnie aktywnej polityki sterowania dostępem.

Wypełnienie pierwszego warunku jest zależne od jakości procedur identyfikacji i uwierzytelniania stanowiących odrębną dziedzinę badań z zakresu bezpieczeństwa SI. Drugi warunek może być wypełniony poprzez włączenie systemowego obszaru danych do procedur objętych sterowaniem dostępem.

Jednym z istotnych problemów związanych ze sterowaniem dostępem jest możliwość zapewnienia zgodności polityki implementowanej na poziomie SI z polityką sterowania dostępem na szczeblu organizacji.

Zgodnie ze standardami z dziedziny ochrony systemów informatycznych wyróżnia się trzy podstawowe polityki sterowania dostępem, które pozwalają modelować politykę organizacji:

- Discretionary Access Control (DAC),
- Mandatory Access Control (MAC),
- Role Based Access Control (RBAC).

Powyższe polityki oraz ich ewentualne modyfikacje mogą być implementowane w oparciu o któryś z trzech następujących mechanizmów sterowania dostępem [Den82]:

- listach dostępu,
- hierarchii dostępu,
- możliwościach.

Zasada działania mechanizmu list dostępu oparta jest na zdefiniowanej w systemie ochrony liście podmiotów wraz z przydzielonymi im prawami do poszczególnych obiektów. Mechanizm hierarchii dostępu bazuje na wyróżnionych w systemie stopniach uprzywilejowania dla podmiotów oraz klasach zaufania dla obiektów. Na podstawie takiej hierarchii podmioty bardziej uprzywilejowane otrzymują zbiór praw będących nadzbiorem praw podmiotów mniej uprzywilejowanych. Możliwości są wyróżnionymi zbiorami kompetencji w systemie, reprezentowanymi poprzez nazwane znaczniki, które są przypisywane podmiotom w zależności od ich bieżących uprawnień.

Sposób wyboru i oceny przydatności poszczególnych mechanizmów sterowania dostępem i ich realizacji jest złożonym zagadnieniem. Jak dotychczas problem ten był omawiany jedynie w kontekście klasyfikowania systemów ze względu na gwarantowany przez nie poziom bezpieczeństwa. Wśród najważniejszych prac poruszających te zagadnienia należy wymienić dokumenty Departamentu Obrony USA (ang. Department of Defence), tzw. *'The Orange Book'* i *'The Red Book'*, które pomimo tego, że powstały na początku lat osiemdziesiątych, jeszcze do niedawna pozostawały jednym z nielicznych źródeł wytycznych odnośnie bezpieczeństwa SI [NCSC85] [NCSC87]. Obok tych publikacji istniała rzadziej stosowana i cytowana europejska wersja ustaleń - *'Information Technology Security Evaluation Criteria'* (ITSEC). Zarówno europejskie jak i amerykańskie dokumenty proponowały w dziedzinie sterowania dostępem wykorzystanie jedynie polityk Discretionary Access Control i Mandatory Access Control wraz ze związanymi z nimi mechanizmami, pomijając rozwiązania oparte na koncepcji ról czyli na polityce Role Based Access Control. Wynikało to przede wszystkim z charakteru organizacji tworzącej normy, a wywodzącej się z kręgów militarnych. Polityki sterowania dostępem Discretionary Access Control i Mandatory Access Control dają zaś duże możliwości sterowania dostępem w organizacjach o prostej i ściśle określonej strukturze hierarchicznej, typowej dla organizacji militarnych, rządowych, itp. Inną przyczyną takiego, a nie innego przesunięcia punktu ciężkości wśród metod sterowania dostępem mogły być czasy, w których wyżej wymienione dokumenty powstawały oraz stopień rozwoju SI w owych czasach. Właśnie ze względu na częściową dezaktualizację oraz na zmianę wymagań stawianych przed SI i ich bezpieczeństwem, w 1996 r. po raz pierwszy została opracowana wspólna dla USA i Europy nowa klasyfikacja bezpieczeństwa systemów, opublikowana pod nazwą *'The Common Criteria'* (CC). Na dzień dzisiejszy obowiązuje wersja druga tych ustaleń z maja 1998 r [ISO99].

Dokument ten, w przeciwieństwie do wcześniejszych systemów klasyfikacji poziomów bezpieczeństwa SI zwraca uwagę na zalety płynące ze stosowania ról jako elementu ochrony. Daje to nowe podstawy do rozwoju badań nad polityką Role Based Access Control, a przede wszystkim możliwość uzyskania przez systemy wspierające tę politykę oficjalnych certyfikatów poświadczających ich zgodność z ogólnie przyjętymi standardami bezpieczeństwa.

Niezależnie od powstania standardu CC, na początku lat dziewięćdziesiątych, grupa naukowców zrzeszona przy National Institute of Standards and Technology (NIST) pracowała nad projektem mającym na celu wypracowanie mechanizmów dla polityk MAC i DAC, pozwalających efektywnie rozwiązać problem sterowania dostępem w złożonych środowiskach systemów rozproszonych. Jako rezultat tych prac powstały podstawowe założenia polityki sterowania dostępem w oparciu o role [SCFY94]. Wkrótce potem powstała formalna specyfikacja wymagań związanych z polityką RBAC oraz pierwsza implementacja systemu sterowania opartego o tę politykę. Wypracowane w NIST założenia po dzień dzisiejszy stanowią podstawę większości opracowań teoretycznych jak i praktycznych implementacji z zakresu sterowania dostępem poprzez role, chociaż jak do tej pory nie zostały one oficjalnie uznane za standard specyfikacji wymagań polityki RBAC.

Założenia przyjęte podczas prac nad polityką RBAC, tzn. potrzeba skutecznego, a zarazem intuicyjnego sterowania dostępem, stanowią o jej atrakcyjności na potrzeby zastosowań we współczesnych organizacjach. Potwierdzeniem tego mogą być prace nad wykorzystaniem RBAC w języku SQL, czy też w systemie CORBA oraz w wielu innych systemach [FGL93] [ISO95] [GC95] [Bar95a] [CORBA95] [SETA96].

1.2. Cel badawczy.

Każda polityka, również polityka sterowania dostępem poprzez role, wymaga wsparcia ze strony mechanizmów sterowania dostępem. Mechanizmami zaś nazywa się środki implementacji polityki w danym systemie informatycznym. Wśród opisanych w literaturze mechanizmów polityki sterowania dostępem poprzez role najważniejsze to:

- a) definiowania ról (rola- {prawa,zakazy,obowiązki}),
- b) przypisywania podmiotów do ról (podmioty-role):
 - stałe,
 - delegowane,
- c) definiowania powiązań pomiędzy rolami (rola-rola):
 - hierarchii ról,
 - ograniczeń:
 - wzajemne wykluczanie się ról,
 - wzajemne wymaganie się ról,
 - ograniczenie na maksymalną liczbę aktywnych podmiotów przypisanych do roli.

Podstawową wadą opracowanych propozycji implementacji polityki Role Based Access Control jest to, iż implementują one jedynie wybrane mechanizmy tej polityki oraz nie pozwalają na automatyzację implementacji mechanizmów. Wśród dostępnych rozwiązań są na przykład takie, które umożliwiają:

- definiowanie ról bez możliwości określenia istniejących powiązań i zależności pomiędzy nimi [Kuh97] [Bar95b] [FBK95]
- definiowanie ról jako zbiorów uprawnień (dozwolonych akcji), bez możliwości jawnego określenia działań zabronionych dla podmiotów odgrywających daną rolę [San97] [SB97] [Gui95] [FK92].

Szersze omówienie problemów związanych z istniejącymi implementacjami polityki Role Based Access Control znajduje się w czwartym rozdziale tej pracy.

Główny problem badawczy podejmowany w pracy dotyczy zatem zagadnień implementacji mechanizmów realizujących politykę Role Based Access Control w rzeczywistych systemach informatycznych.

Teza prezentowanej rozprawy sformułowana jest w następujący sposób:

TEZA:

Istnieje możliwość opracowania modelu mechanizmów polityki sterowania dostępem poprzez role umożliwiającego automatyzację ich implementacji.

Udowodnienie powyżej sformułowanej tezy wiąże się z koniecznością realizacji następującego celu badawczego:

CEL BADAWCZY:

Zaproponowanie i opracowanie modelu mechanizmów sterowania dostępem poprzez role oraz charakterystyka własności opracowanego modelu.

Z powyższym celem wiązą się następujące zadania:

- Zaproponowanie modelu mechanizmów polityki sterowania dostępem poprzez role,
- Analiza własności opracowanego modelu,
- Opis mechanizmów polityki sterowania dostępem poprzez role:
 - opracowanie algorytmów,
 - sformułowanie i udowodnienie twierdzeń o właściwościach prezentowanych algorytmów,

Na podstawie krytycznej analizy czterech wybranych implementacji polityki Role Based Access Control (rozdz. 4.) [Kuh97] [FBK95] [San97] [SB97], określono następujące cechy formalnego opisu mechanizmów polityki sterowania dostępem poprzez role:

- Zgodność z podstawowymi paradygmatami polityki sterowania dostępem poprzez role, tzn.:
 - intuicyjność zapisu polityki,
 - możliwość zapisu polityk dla różnorodnych organizacji,
- Możliwość implementacji mechanizmów polityki sterowania dostępem poprzez role:
 - a) definiowania ról (rola- {prawa,zakazy,obowiązki}),
 - b) przypisywania podmiotów do ról (podmioty-role):
 - stałe,
 - delegowane,
 - c) definiowania powiązań pomiędzy rolami (rola-rola):
 - hierarchii ról,
 - ograniczeń (ang. constraints):
 - wzajemne wykluczanie się ról,
 - wzajemne wymaganie się ról,
 - ograniczenie na maksymalną liczbę aktywnych podmiotów przypisanych do roli.
- Możliwość automatyzacji implementacji mechanizmów polityki sterowania dostępem poprzez role.

Cel badawczy pracy wraz ze związanymi z nim zadaniami realizowany jest w następujący sposób:

- Przedstawiono krytyczną analizę istniejących mechanizmów polityki Role Based Access Control,
- Nawiązano do istniejących prób wstępnej formalizacji ról w oparciu o formalizm logiki deontycznej [CC97] [CC98] [Cup94] [RF98].

Istotą tych rozważań jest przedstawienie założeń polityki Role Based Access Control w postaci formuł logicznych i opracowanie systemu aksjomatycznego opartego w szczególności na aksjomatach i regułach inferencji systemów deontycznych. Uznano tym samym zalety dotychczasowego sposobu interpretowania pojęcia roli w systemach deontycznych.

- Skonstruowano model RMM (RBAC Modal Model) dla polityki sterowania dostępem poprzez role, a w tym:
 - zdefiniowano język modelu,
 - sformułowano warunki korzystania z języka modelu,
 - przedstawiono analizę własności opracowanego modelu, a w tym:
 - udowodniono możliwość wykorzystania translacji semi-funkcyjnej [BDRS98],
 - zdefiniowano algorytmy automatyzacji procesu weryfikowania poprawności zapisu i modyfikacji polityki
 - opracowano metodę opisu mechanizmów polityki kontroli dostępu:
 - definiowania ról:
 - stałe,
 - delegowane,
 - przypisywania podmiotów do ról,
 - definiowania powiązań pomiędzy rolami,
 - hierarchii ról,
 - ograniczeń.
 - podano przykłady ilustrujące możliwości opracowanego modelu.

Jednym z elementów na drodze realizacji celu badawczego jest także prezentacja możliwości praktycznej implementacji sterowania dostępem w postaci programów logicznych języka PROLOG

Osiągnięte rezultaty jako całość stanowią realizację postawionego celu badawczego oraz dowodzą zasadności tezy dla rozważanego problemu.

1.3. Porządek pracy.

Rozdz. 1 - Wstęp.

Zawiera krótkie wprowadzenie w tematykę pracy oraz określenie celu badawczego. W tej części pojawia się pierwsze sformułowanie głównego zakresu badawczego pracy oraz jej tezy, która zostaje udowodniona w dalszych rozdziałach.

Rozdz. 2 - Bezpieczeństwo systemów informatycznych.

W rozdziale drugim określony jest zakres tematyczny pracy oraz podane zostają definicje podstawowych pojęć z tego zakresu, tzn. związanych z bezpieczeństwem systemów informatycznych i sterowania dostępem. W oparciu o istniejące standardy przedstawiono elementy składowe sterowania dostępem takie jak identyfikacja, uwierzytelnienie, autoryzacja oraz logiczne sterowanie dostępem oraz scharakteryzowano istniejące pomiędzy nimi zależności. Uzasadniono zawężenie późniejszych rozważań jedynie do obszaru mechanizmów sterowania dostępem.

Rozdz. 3 - Przegląd podstawowych polityk sterowania dostępem.

Rozdziale trzecim zawarta jest charakterystyka podstawowych rozwiązań w zakresie sterowania dostępem. Omawiam dwie polityki Discretionary Access Control i Mandatory Access Control wraz z powiązanymi z nim mechanizmami. Zamieszczony w pracy opis tych polityk służy dwóm celom. Po pierwsze, określeniu szerszego kontekstu dla prezentowanych rozważań. Wyżej wymienione polityki stanowiły przez wiele lat podstawę wszystkich systemów związanych z sterowaniem dostępem, a polityka Role Based Access Control, która jest głównym przedmiotem zainteresowania tej pracy, powstała zaś w wyniku badań nad możliwością usprawnienia sterowania dostępem tych polityk. Po drugie, określeniu domeny zastosowań dla polityki Role Based Access Control. Polityki tej, pomimo większych od swoich poprzedniczek możliwościach modelowania polityk sterowania dostępem dla organizacji o różnorodnych profilach, nie należy traktować jako rozwiązania, które całkowicie wyparło możliwość stosowania polityk Discretionary Access Control i Mandatory Access Control. Polityka Discretionary Access Control ma np. przewagę nad pozostałymi politykami w przypadku organizacji, w których sterowanie dostępem odbywa się w oparciu o prawo własności obiektów [NCSC87a]. Natomiast Mandatory Access Control najlepiej odzwierciedla zależności panujące w organizacjach o prostej i ścisłej hierarchicznej strukturze, jaką posiadają np. organizacje militarne [Lee88] [NCSC85].

Rozdz. 4 - Polityka i mechanizmy RBAC.

Pierwszą część rozdziału czwartego stanowi charakterystyka polityki sterowania dostępem poprzez role (Role Based Access Control - RBAC). Przedstawione założenia odnośnie polityki RBAC stanowią jedno z dwóch źródeł założeń wstępnych przedstawionego w rozdziale piątym modelu mechanizmu tej polityki. Kolejna część tego rozdziału zawiera krytyczny przegląd istniejących rozwiązań odnośnie mechanizmów wspierających sterowanie dostępem poprzez role. Przedstawiono cztery przykładowe implementacje tych mechanizmów, które powstały jako projekty w głównym ośrodku zajmującym się badaniami nad polityką RBAC w latach dziewięćdziesiątych, czyli w National Institute of Standards and Technology (NIST). Jednym ze szczegółowo omówionych modeli są modele RBAC'96 i

ARBAC'97, które uważa się za najlepsze opracowanie określające wymagania i cechy funkcjonalne polityki RBAC. W omówieniach poszczególnych mechanizmów zawarta jest lista nierozwiązanych przez dany mechanizm problemów. Na podstawie tego przeglądu, precyzuję listę wymagań koniecznych dla osiągnięcia w pełni funkcjonalnego mechanizmu dla polityki RBAC. Lista ta jest źródłem wymagań dla tworzonego w piątym rozdziale mechanizmu.

Rozdz. 5 - Model sterowania i zarządzania dostępem oparty o politykę RBAC.

W rozdziale piątym przedstawiono model *RMM* (RBAC Modal Model) stanowiący formalną podstawę mechanizmów zarządzania i sterowania dostępem poprzez. Na początku tego rozdziału przedstawiono dyskusję nad normatywnym aspektem sterowania dostępem. Na podstawie tych rozważań i w oparciu o dotychczasowe zastosowania logiki deontycznej, uzasadniono wybór tej logiki jako podstawy formalizmu dla tworzonego mechanizmu. Określono język opisu polityki oraz zasady jego wykorzystania na potrzeby zapisu polityki kontroli dostępu poprzez role.

Rozdz. 6 - Mechanizm sterowania dostępem.

Rozdział szósty zawiera algorytm analizy spójności żądań podmiotów z przyjętą polityką oraz dowód twierdzenia o dopuszczeniu do wykonania przez podmiot operacji dostępu. Algorytm ten wraz z twierdzeniem stanowią podstawę mechanizmu umożliwiającego weryfikację zgodności zgłaszanych przez podmioty żądań dostępu z obowiązującą w systemie polityką sterowania dostępem.

Rozdz. 7 - Mechanizmy zarządzania dostępem.

W rozdziale siódmym w oparciu o opracowany wcześniej model *RMM* przedstawiony został opis mechanizmów zarządzania dostępem takich jak definiowanie powiązań typu rola-rola oraz powiązań rola-prawa dostępu i rola-podmiot.

Na potrzeby opisu wyżej wymienionych mechanizmów opracowano dwa algorytmy: badania spójności nowododawanej formuły z istniejącym zbiorem formuł i budowania hierarchii ról. Sformułowano i udowodniono również twierdzenia o spójności nowododawanej formuły z istniejącym zbiorem formuł, o ograniczeniu dziedziczenia uprawnień, o spójności hierarchii ról, o spójności hierarchii ról i o wzajemnym wymaganiu się ról.

Rozdz. 8 - Podsumowanie.

W posumowaniu dokonano przeglądu najważniejszych wyników oraz wskazania kierunku dalszej pracy.

Dodatki.

- Dodatek A.** - Wykaz ważniejszych skrótów wykorzystanych w pracy,
Dodatek B. - Podstawowe twierdzenia z zakresu matematyki i logiki,
Dodatek C. - Przegląd najważniejszych elementów zdefiniowanego mechanizmu sterowania dostępem.
Dodatek D. - Bibliografia.

1.4. Odniesienia literaturowe.

Wśród źródeł, które były wykorzystane przeze mnie podczas powstawania tej pracy można wyróżnić sześć zasadniczych kategorii związanych z następującymi tematami:

1. Bezpieczeństwo systemów informatycznych.
2. Logiczne sterowanie dostępem.
3. Źródła dotyczące polityki RBAC.
4. Zastosowania logiki matematycznej w sterowaniu dostępem.
5. Logika matematyczna
6. Pozostałe.

ad. 1.)

Literatura związana z tym tematem zawiera przede wszystkim podstawowe wiadomości z zakresu bezpieczeństwa systemów informatycznych, takie jak standardy, definicje pojęć, formalizmy, itp. Do podstawowych opracowań z tego zakresu należy zbiór uznanych standardów [ISO99] ISO/IEC 15408, *Common Criteria*, v.2.0 oraz tzw. standardów de facto opracowanych przez DoD [NCSC85] *Department of Defence Trusted Computer Security Evaluation Criteria*, i [NCSC87] *Trusted Network Interpretation*. Pozostałe publikacje zawierające usystematyzowane wiadomości z tej kategorii to, [Bib77] *Integrity Considerations for Secure Computer Systems*, [GE88] *Reasoning about knowledge and permission in secure distributed systems*, [GM84] *Security Policies and Security Models*, [GM84] *Unwinding, Inference Control*, [NCSC92] *A Guide to Understanding Security Modelling in Trusted Systems*, [Lea94] *Security models*

ad. 2.)

Literatura odnosząca się do szeroko pojętego logicznego sterowania dostępem zawiera wymagania polityk, przykłady zastosowań, problemy związane z poszczególnymi politykami sterowania dostępem, itp. Np. [NCSC87a] *A Guide to Understanding Discretionary Access Control In Trusted Systems*, jest jednym z podstawowych dokumentów definiujących podstawy polityki DAC. Analogicznie dla MAC istotne są dokumenty [CW87] *A Comparison of Commercial and Military Computer Security Policies*, [BK85] *A practical alternative to hierarchical integrity policies.*, [San92] *Lattice-based enforcement of chinese*

walls.[San93] *Lattice-based access control models*. Pozostałe dokumenty z tego zakresu to [Bib77] *Integrity Considerations for Secure Computer Systems*, [BN89] *The Chinese wall security policy* [Car83] *Permissions and obligations*. [Cup93] *A Logical Analysis of Authorized, Prohibited Information Flows*. [Den74] *A Lattice Model of Secure Information Flow*, [Den76a] *On the Derivation of Lattice structured Information Flow Policies* [Den77] *Certification of Programs for Secure Information Flow*, [Den82] *Cryptography and Data Security*, [Den86] *Views for Multilevel Database Security*, [FGL93] *An Examination of Federal and commercial Access Control Policy Needs*, [JK99] *Analysis and Evaluation of Network Security Policies* [Kol00b] *Effective Access Control* [Lam74] *Protection*. [LHL74] *A Security Model for military Message Systems*, [Lea90] *Security models and information flow*. [Lee88] *Using mandatory integrity to enforce commercial security*. [Loz91] *Integrity in Automated Information Systems*. [LS90] *A Model for Multilevel Security in Computer Networks*, [Lun88] *Access control policies: Some unanswered questions*. [MS91] *Delegation of Authority*. [MS93] *Policy hierarchies for distributed systems management*. [MS94] *Policy conflict analysis in distributed system management*. [SS94] *Access Control: Principles and Practice*, [San88] *Transaction control expressions for separation of duties*. [San90] *Separation of duties in computerized information systems*. [SS97] *Authentication, access control and intrusion detection*.

ad. 3.)

W materiałach źródłowych odnośnie polityki RBAC znajduje się wiedza na temat postępu prac nad tą polityką, przykłady zastosowań, analiza problemów, etc. Do tej kategorii zaliczono następujące pozycje: [Bar95b] *Implementing Role-Based Access Control using Object Technology*, [BCFGK97] *Role based Access Control for the World Wide Web*. [FB97] *Specifying and Managing Role-Based Access Control within a Corporate Intranet*, [FCK95] *Role-Based Access Control (RBAC): Features and Motivations* [FK95] *Future Directions in Role-Based Access Control* [FK92] *Role-Based Access Controls*, [Gui95] *A new model for role-based access control*, [Kol00a] *Zarządzanie dostępem do systemu informatycznego poprzez role*. [Kuh97] *Mutual Exclusion of Roles as a Means of Implementing Separation of Duty in RBAC Systems* [NO95] *Access rights administration in rolebased security systems* . [SCFY94] *Role-based access control: A multi-dimensional view*. [Tho91] *Rolebased application design and enforcement* [San96] *Role hierarchies and constraints for lattice-based access controls*. [San97] *Rationale for the RBAC96 family of access control models*. [San98] *Role activation hierarchies*. [SB97] *The URA97 model for role-based administration of user-role assignment* [SBCG97] *The ARBAC97 model for role-based administration of roles: preliminary description and outline*. [SBM99] *The ARBAC97 model for role-based administration of roles* [SM98] *How to do discretionary access control using roles*.

ad. 4.)

W dziedzinie zastosowań logiki matematycznej w sterowaniu dostępem znajdują się pozycje analizujące możliwości wykorzystania formalizmu logiki do analizy sterowania dostępem od teoretycznych rozważań, aż po próby implementacji konkretnych modeli. Traktują o tym następujące publikacje: [ABN89] *A logic of authentication*, [BC91a] *Computer Security Policies, Deontic Logic*. [BC91b] *A Definition of Secure Dependencies Using the Logic of Security* [BC92] *A Logical View of Secure Dependencies* [BC93] *Expression of confidentiality policies with deontic logic* [CC95] *Solving normative conflicts by merging roles*. [CC97] *Analyzing Consistency of Security Policies*, [CD96] *A Deontic Logic for Reasoning about Confidentiality*. [CS96] *Specyfing a Security Policy: A Case Study*, [Cup94] *Roles and Deontic Logic*. [Cup95] *A Normative Framework for Security Policies*, [FT98] *Formal models of fraud*, [GE90] *A logic for reasoning about security*. [GGF98] *On the Formal Definition of Separation-of -Duty Policies and their Composition* [Cho97] *An application of SOL-resolution: checking regulation consistency*, [Kol98] *Modelling Management Goals and Policies with Deontic Logic* [Kol99a] *Analysis of Network Security Policy Consistency*, [Kol99b] *Analiza polityk bezpieczeństwa z wykorzystaniem logik modalnych*. [Lee67] *A completeness theorem and computer program for finding theorems derivable from given axioms*. [RF98] *A deontic logic for diagnosis of organisational process design*. [Sut86] *A Model of Information*.



ad. 5.)

Pozycje odnośnie logiki matematycznej wiążą się z takimi tematami jak logika modalna, logika deontyczna, metody automatyzacji wnioskowania. Należą do nich: [Bat94] *Podstawy logiki*, [BDRS95] *Wnioskowanie w logikach nieklasycznych. Podstawy teoretyczne*. [BDRS98] *Wnioskowanie w logikach nieklasycznych. Automatyzacja wnioskowania*. [Bor77] *Logika formalna* [God67] *On formally undecidable propositions of principia mathematica and related systems*. [Grz84] *Zarys logiki matematycznej*, [Gum90] *Wprowadzenie w logikę współczesną*, [Hil81] *Deontic Logic: Introductory and Systematic Readings*, [Kan72] *Law and Logic*. [Kri63] *Semantical analysis of modal logic I normal modal prepositional calculi*, [Kul99] *Logika programowania, a sylogistyka Arystotelesa* [Mar87] *Logika formalna. Zarys encyklopedyczny z zastosowaniem do informatyki i lingwistyki* [PoS70] *O dowodzie matematycznym*, [PSu96] *Elementarny wykład logiki formalnej z ćwiczeniami komputerowymi* [RL91] *Defeasible deontic reasoning : a logic programming model*. [Suc80] *Studia nad logiką deontyczną*, [Zie68] *Paradoksy logiki deontycznej*, [Zie69] *Logika deontyczna jako formalizacja rozumowań normatywnych*,

ad. 6.)

Do kategorii pozostałe zaliczone zostały przede wszystkim opracowania dotyczące praktycznych zastosowań związanych z omawianą w pracy teorią. Są to następujące pozycje: [Bar95a] *Application Engineering in Health Care*, [CC98] *Reasoning about norms provided by conflicting regulations*, [CORBA95] *CORBA Security Draft*, [GC95] *A Strategy for*

of the Electronic Patient Record, [Ino91] *Consequencefinding based on ordered linear resolution*. [ISO95] ISO/IEC 9075, Database Language SQL - Part 2: Foundation [JS93] *On the characterization of law and computer systems: the normative systems perspective*. [Kol97] *The advantages of Distributed Management* [Loi94] *Refinement of knowledge bases based on consistency*. [OL96] *A decision support system for bureaucratic policy administration: abductive logic programming approach* [Por77] *Action Theory, Social Science; Some Formal Models*, [PS94] *ContrarytoDuty Imperatives, Defeasibility, Violability* . [SETA96] SETA [SM94] *The management of computer security profiles using a roleoriented approach* . [WM93] *Applications of deontic logic to computer science: a concise overview*.

2. Bezpieczeństwo systemów informatycznych.

2.1. Elementy bezpieczeństwa systemów informatycznych.

Celem bezpieczeństwa systemu informacyjnego (SI) jest ochrona wartościowych zasobów systemu takich jak przetwarzane w nim informacje, dane, wchodzący w jego skład sprzęt i oprogramowanie. Dzięki wyborowi i zastosowaniu odpowiednich zabezpieczeń bezpieczeństwo SI wspiera realizację misji organizacji poprzez ochronę jej fizycznych, finansowych zasobów, czy też ochronę marki organizacji, jej pozycji prawnej, itp. Często jednak sprawa zapewnienia bezpieczeństwa SI jest postrzegana jako przeszkoda na drodze szybkiego i efektywnego działania. Ma to miejsce w przypadku, gdy zastosowano niewłaściwe środki ochrony, które kojarzą się jedynie z dodatkowymi wymogami nałożonymi na pracowników i administratorów systemu, nie dostarczając w zamian żadnych zauważalnych korzyści. Przy projektowaniu i wdrażaniu systemu bezpieczeństwa należy wystrzegać się takiego stanu rzeczy. Należy pamiętać o tym, iż dobry system bezpieczeństwa nie funkcjonuje sam dla siebie, jest zaś wprowadzany jedynie po to aby wspierać główną misję organizacji. Podsumowując, bezpieczeństwo jest środkiem prowadzącym do celu, nie zaś celem samym w sobie [GM84a] [GM84b] [NCSC92].

Jak do tej pory nie powstała jednolita, przyjęta przez wszystkich definicja bezpieczeństwa SI. Większość z pojawiających się w literaturze definicji posiada jednak wiele zbieżnych ze sobą elementów. Poniżej zamieszczono trzy przykładowe definicje tego terminu.

1. **Bezpieczeństwo SI** - są to środki ochrony SI dostarczone w celu uzyskania pożądaných rezultatów odnośnie zachowania integralności, dostępności oraz poufności zasobów. Dotyczy to zarówno sprzętu jak i oprogramowania, danych/informacji, możliwości transmisyjnych itd. [NCSC92]
2. **Bezpieczeństwo SI** - jest to zbiór miar oraz sposobów sterowania, które pozwalają ochronić SI przed odmową usługi, nieautoryzowanym ujawnieniem (przypadkowym jak i zamierzonym) informacji, modyfikacją lub uszkodzeniem komponentów SI i jego danych. Bezpieczeństwo SI odnosi się do wszystkich jego sprzętowych lub/i programowych funkcji, procedur operacyjnych, sterowania dostępem do jednostki centralnej, odległych stacji, terminali, ograniczeń zarządzania, struktury fizycznej i urządzeń, międzyosobowych i komunikacyjnych sposobów sterowania koniecznych dla zapewnienia akceptowalnego poziomu ryzyka dla SI oraz jego danych. [Lea94]
3. **Bezpieczeństwo SI** - celem jego jest ochrona systemu przed niewłaściwym użyciem jego programów i danych składowanych. Bezpieczeństwo powinno gwarantować spełnienie wymagań odnośnie poufności, integralności i dostępności systemu [GM84].

W celu utworzenia jak najbardziej ogólnego modelu bezpieczeństwa wspólnego dla wszystkich SI, a jednocześnie dającego możliwość precyzyjnego opisu warunków bezpieczeństwa, wyróżnia się kilka poziomów opisu. Pozwalają one przejść od najbardziej ogólnej charakterystyki bezpieczeństwa na poziomie organizacji jako całości, poprzez wyróżnienie odgrywanej przez SI roli, aż do poziomu opisu technicznych środków zaangażowanych w ochronę bezpieczeństwa.

W zbiorze dokumentów *Common Criteria* stanowiących międzynarodowy standard z zakresu oceny poziomu bezpieczeństwa systemów informatycznych, wyróżniono następujące poziomy opisu bezpieczeństwa [ISO99]:

- *bezpieczeństwo środowiska (Security Environment)*
Poziom ten opisuje prawa, polityki bezpieczeństwa organizacji, definiujące kontekst w którym będzie działał oceniany system oraz zagrożenia wnoszone przez środowisko jego pracy.
- *cele bezpieczeństwa (Security Objectives)*
Zawiera te spośród zdefiniowanych zagrożeń, polityk bezpieczeństwa, które będą brane pod uwagę przy implementacji systemu bezpieczeństwa.
- *wymagania bezpieczeństwa (TOE Security Requirements)*
Przekształcone cele bezpieczeństwa do poziomu technicznych wymagań na potrzeby funkcji bezpieczeństwa i ochrony, pokrywających oceniany system i jego środowisko informatyczne.
- *specyfikacja bezpieczeństwa (TOE Security Specification)*
Definiuje aktualną lub proponowaną implementację ocenianego systemu.
- *implementacja (TOE Implementation)*
Realizacja ocenianego systemu w zgodzie z przyjętą specyfikacją.

Zgodnie z przedstawioną powyżej hierarchią, pierwszym krokiem podczas opracowywania systemu bezpieczeństwa powinno być rozpatrzenie wymogów i zależności pochodzących od środowiska zewnętrznego względem SI. Podstawowym środowiskiem pracy każdego systemu jest organizacja w której on funkcjonuje oraz organizacje z nią współpracujące. Wymagania i zależności pochodzące od środowiska wynikają więc bezpośrednio z polityki organizacji [JK99]. Dlatego na podstawie analizy polityki organizacji powinna zostać utworzona polityka bezpieczeństwa.

Na potrzeby dalszych rozważań, przyjęto następującą definicję:

Polityka bezpieczeństwa - zbiór praw, reguł, praktyk, które regulują sposób wykorzystania informacji łącznie z jej przetwarzaniem, przechowywaniem, dystrybucją i prezentacją.

Polityka bezpieczeństwa określa więc zasady sterowania rozprzestrzenianiem się, modyfikowaniem i dostępem do informacji. Musi ona być precyzyjnie zdefiniowana, zaimplementowana dla wszystkich systemów przetwarzających informacje wrażliwe pod względem bezpieczeństwa oraz dokładnie odzwierciedlać prawa, uregulowania i ogólne polityki na podstawie których jest definiowana [NCSC85].

Zawarty w polityce bezpieczeństwa zbiór reguł i praw wyznacza zakres usług koniecznych do zaimplementowania w systemie. Architektura bezpieczeństwa OSI wyróżnia pięć klas usług bezpieczeństwa [NCSC92]

1. Uwierzytelnianie.
2. Poufność danych.
3. Integralność danych.
4. Dostępność.
5. Sterowanie dostępem.

ad.1)

Kompletny proces poprawnego uwierzytelniania użytkowników systemu składa się z etapu identyfikacji, uwierzytelnienia i autoryzacji.

Identyfikacja (ang. identification) polega na możliwości rozróżnienia wielu kategorii użytkowników systemu, a przez to umożliwia się tworzenie w systemie dedykowanych profili. Profile są podzbiorami zbioru wszystkich możliwych do przeprowadzenia operacji w systemie, dostosowanymi do indywidualnych potrzeb i wymagań użytkownika korzystającego z danego profilu. Identyfikacja nie zakłada możliwości weryfikacji czy użytkownik zgłaszający chęć do skorzystania z określonego profilu jest rzeczywiście tym za kogo się podaje i czy w związku z tym rzeczywiście powinno mu przysługiwać prawo dostępu do danego profilu.

Weryfikacją tożsamości użytkowników w systemie informatycznym zajmuje się proces uwierzytelniania (ang. authentication). Ma on za zadanie potwierdzenie tego, że użytkownik zgłaszający chęć pracy w systemie jest rzeczywiście tym, za kogo się podaje. W tym celu stosuje systemy haseł, kart magnetycznych, badań biometrycznych, itp.

Po potwierdzeniu tożsamości użytkownika następuje jego autoryzacja (ang. authorization). Autoryzacja jest procesem dzięki któremu system określa na podstawie wcześniej określonej tożsamości, zbiór operacji dostępnych dla danego użytkownika [GE88], [Lam74], [LHL74], [MS91].

ad 2.)

Poufność (ang. privacy) jest elementem systemu bezpieczeństwa odpowiedzialnym za zapewnienie prywatności gromadzonych, przetwarzanych i przesyłanych w systemie danych, jak i również aktywności użytkownika w systemie. System powinien gwarantować każdemu użytkownikowi, że nikt niepowołany do tego nie będzie mógł wejść w posiadanie danych nie przeznaczonych dla niego, ani też nie będzie mógł zdobyć nadmiernej wiedzy o danych przetwarzanych w systemie i o aktywności innych użytkowników. Podstawowym mechanizmem związanym z ochroną poufności w systemie są metody kryptograficzne [Den82], [Den77].

ad 3.)

Integralność (ang. integrity) ma na celu uniemożliwić zmianę danych przetwarzanych, przesyłanych i składowanych w systemie w nieuprawniony sposób. W procesie ochrony integralności systemu, stosuje się metody umożliwiające wykrywanie wprowadzanych zmian np. poprzez zapamiętywanie tożsamości użytkownika dokonującego modyfikacji zasobów systemu [CW87].

ad 4.)

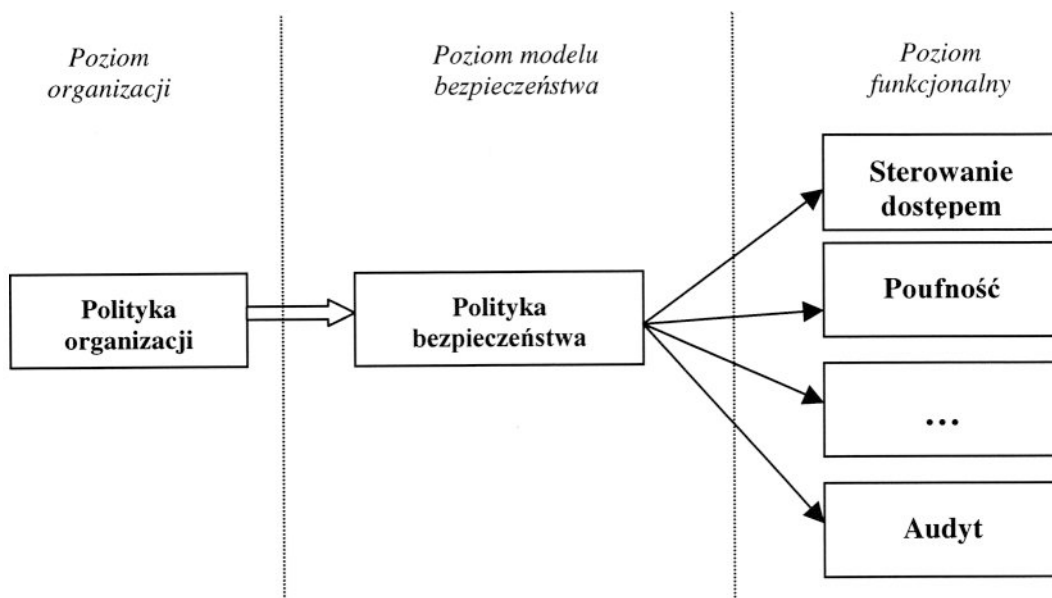
Dostępność (ang. availability) systemu, jest związana z żądaniem, aby system gwarantował wykonanie każdej dozwolonej dla danego użytkownika operacji w z góry ustalonym przedziale czasu [BC91b].

ad 5.)

Sterowanie dostępem (access control) w najbardziej ogólnym znaczeniu nadzoruje wszystkie wykonywane operacje w systemie. Szersze omówienie znaczenia sterowania dostępem dla systemów informatycznych znajduje się w dalszej części tego rozdziału.

W dokumentach *Common Criteria* dokonano nieco innego rozróżnienia usług systemów bezpieczeństwa. Wyróżnionych jest tam aż dziewięć obszarów, które mają za zadanie scharakteryzować najbardziej istotne elementy ochrony systemów informatycznych Są to jak następuje [ISO99]:

audyt, komunikacja, ochrona danych użytkownika, wsparcie kryptograficzne, identyfikacja i uwierzytelnienie, zarządzanie bezpieczeństwem, zaufane ścieżki, wykorzystanie źródeł, prywatność, dostęp, ochrona funkcji bezpieczeństwa.



rys.2.1. Elementy procesu definiowania systemu ochrony SI

Oprócz poprawnego zdefiniowania funkcji dla wszystkich wyżej wymienionych obszarów bezpieczeństwa, równie ważny dla prawidłowego funkcjonowania całości, jest najniższy poziom, czyli poziom mechanizmów bezpieczeństwa. Mają one za zadanie realizację wyróżnionych i zdefiniowanych na wyższych poziomach usług funkcji systemu bezpieczeństwa. Na poziomie mechanizmów mamy możliwość wyboru technicznej realizacji określonego przez politykę celu.

Każda funkcja bezpieczeństwa może korzystać z różnorodnych mechanizmów umożliwiających wybór pomiędzy rozwiązaniami różniącymi się poziomem wymagań nakładanych na system czy też poziomem zapewnianego bezpieczeństwa.

Mechanizmami są np. metody szyfrowania z kluczami publicznymi czy też tajnymi wraz z odpowiednimi dla nich algorytmami szyfrowania i służą one do implementacji funkcji z zakresu ochrony poufności systemu.

2.2. Sterowanie dostępem.

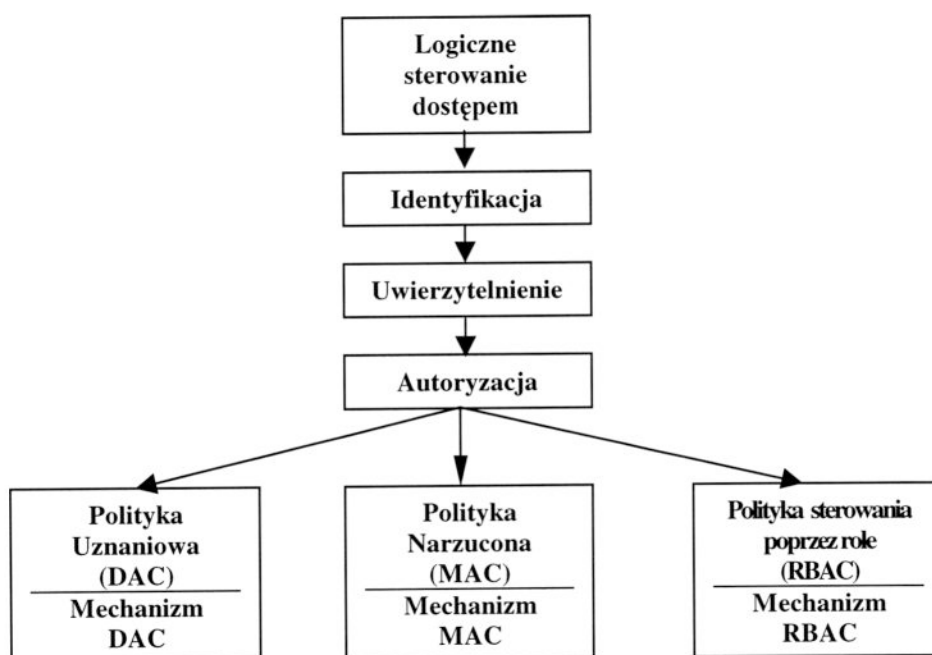
Jedną z bardziej istotnych usług bezpieczeństwa wielodostępnych systemów informatycznych jest sterowanie dostępem do jego zasobów. Poniżej przedstawiono trzy definicje terminu *sterowania dostępem* za [NCSC92]

- Sterowanie dostępem**
- Proces ograniczający dostęp do zasobów systemu informatycznego jedynie autoryzowanym użytkownikom, programom, procesom, systemom lub innym elementom SI. Synonim z dostępem sterowanym, dostępem ograniczonym.
 - Ograniczenie praw lub możliwości podmiotów do komunikowania się z innymi podmiotami lub do aktywacji funkcji i usług w systemie komputerowym lub sieci.
 - Ograniczenie sterujące dostępem podmiotów do obiektów.

Jak wynika z powyższych definicji, termin *sterowanie dostępem* może być rozumiany na wiele sposobów. Jest to powodowane tym, iż dla pełnego bezpieczeństwa systemu, równie ważna jest możliwość sterowania dostępem do jego zasobów na poziomie fizycznym, tzn. dostępu do pomieszczeń, łączy danych, itp. jak i na poziomie logicznym tzn. sterowania interakcjami pomiędzy podmiotami i obiektami systemu czy też sterowaniem przepływami informacji. Dlatego w niektórych publikacjach wprowadza się dodatkowo pojęcie logicznego sterowania dostępem (Logical Access Control - LAC) w celu rozróżnienia sterowania dostępem na różnych poziomach abstrakcji [NCSC92].

W skład systemu logicznego sterowania dostępem wchodzi następujące procesy:

- identyfikacji,
- uwierzytelnienia,
- autoryzacji,
- sterowania odwołaniami do zasobów systemu w zgodzie z przyjętą polityką.



rys.2.2. Elementy logicznego sterowania dostępem.

Skuteczność na najniższym poziomie LAC jest w związku z tym uzależniona zarówno od wiarygodności sterowania dostępem ponad poziomem logicznego sterowania, jak i od poprawności realizacji procedur wstępnych wchodzących w zakres LAC. Do najbardziej istotnych warunków poprawności realizacji procesu sterowania dostępem można zaliczyć:

- konieczność zapewnienia fizycznej ochrony, zwłaszcza w odniesieniu do najbardziej newralgicznych elementów systemu, takich jak serwery, macierze dyskowe, itp.,
- konieczność zapewnienia poprawnej identyfikacji, uwierzytelnienia i autoryzacji,
- konieczność zagwarantowania bezpieczeństwa przechowywanym w systemie regułom sterowania logicznym dostępem, na podstawie których mechanizmy sterowania wypracowują decyzje o zezwoleniu na dostęp.

Jeżeli nie da się zagwarantować możliwości osiągnięcia wszystkich powyżej wymienionych celów, nawet najlepszy system LAC nie będzie w stanie całkowicie wyeliminować możliwości popełniania nadużyć przez nieuprawnionych użytkowników czy też błędów dostępu. Dlatego, na potrzebę dalszych rozważań przyjęto, że system spełnia wszystkie powyżej wymienione warunki wstępne.

Pośredni wpływ na jakość LAC mogą mieć również i inne elementy funkcjonowania systemu, takie jak możliwość zapewnienia poufności w systemie, jego integralność, itp. W niniejszej pracy nie zajmowano się problemami tego typu, gdyż wykraczają one poza jej zakres.

Pojęciami pierwotnymi dla LAC jest pojęcie obiektu, podmiotu, dostępu, typu dostępu, operacji. Będą one pojawiać się w dalszej części pracy w kontekście zgodnym z poniższymi definicjami [NCSC92]:

Podmiot	- aktywny element systemu, który może reprezentować użytkownika, procesy, urządzenia. Aktywność podmiotów może powodować przepływ informacji pomiędzy obiektami, bądź wpływać na zmianę stanu systemu. Podmiot może również pełnić funkcję obiektu.
Obiekt	- pasywny element systemu, który zawiera, bądź otrzymuje informacje. Dostęp do obiektu zazwyczaj implikuje dostęp do składowanej w nim informacji. Przykładami obiektów mogą być rekordy, bloki, segmenty pamięci, pliki, programy.
Typ dostępu	- akcje które mogą być wykonywane względem określonego obiektu przez podmioty systemu, takie jak odczyt, zapis, modyfikacja, usunięcie, utworzenie, itp.
Dostęp	- (do informacji) zdolność wraz z możliwością uzyskania wiedzy o sklasyfikowanej lub niesklasyfikowanej informacji, - (do zasobów) typ interakcji pomiędzy obiektem a podmiotem, którego efektem jest przepływ informacji pomiędzy nimi
Transakcja	- ciąg elementarnych akcji dostępu tworzących logiczną całość na wyższym poziomie abstrakcji opisu działania systemu.

Opierając się na powyżej przedstawionych pojęciach, podstawowe zadanie logicznego sterowania dostępem możemy zdefiniować jako sposób ograniczenia dostępu podmiotów do obiektów. Uwzględniając możliwości podmiotów jakie posiadają one w zakresie interakcji pomiędzy sobą nawzajem oraz do aktywacji funkcji systemu, można do zadań LAC dodać również ograniczenie praw lub możliwości podmiotu do komunikowania się z innymi podmiotem lub do aktywacji funkcji bądź usług w SI (patrz def. sterowania dostępem 2 i 3). Zbiór dostępnych dla danego podmiotu operacji, akcji dostępu, powinien być określony na podstawie jego tożsamości ustalonej w procesie identyfikacji i uwierzytelnienia.

Ostateczna decyzja o akceptacji, bądź o odrzuceniu żądania dostępu leży w zakresie kompetencji mechanizmów sterowania dostępem. Mechanizmy są elementami zaprojektowanymi w celu wykrywania i uniemożliwienia nieautoryzowanego dostępu oraz zezwalającymi na dostęp autoryzowany. Mogą one być implementowane zarówno w postaci dodatkowych urządzeń technicznych, bądź też w postaci procedur będących elementem jądra systemu operacyjnego, czy też programów działających w warstwie aplikacji.

2.3. Zarządzanie dostępem.

Zarządzanie dostępem polega na dokonywaniu zmian autoryzacji podmiotów do podejmowania określonych akcji dostępu. Odbywa się ono poprzez dodawanie podmiotom nowych i usuwanie przyznanych wcześniej uprawnień, etc. Zarządzanie dostępem odbywa się w oparciu o mechanizmy skojarzone z daną polityką sterowania dostępem. W związku z tym, dla polityki uznaniowego sterowania dostępem (DAC) możliwości zarządzania dostępem będą związane z przyjętym sposobem zarządzania prawem sterowania. W przypadku polityki narzuconej (MAC) zarządzanie dostępem ogranicza się do etapu klasyfikowania podmiotów i obiektów systemu przez jego administratora. Polityka sterowania dostępem poprzez role (RBAC) zarządza dostępem na kilku poziomach. Zarządzanie dostępem w tej polityce jest związane z mechanizmem definiowania powiązań ról z prawami dostępu, mechanizmem definiowania powiązań pomiędzy rolami i podmiotami. oraz z mechanizmem definiowania powiązań pomiędzy rolami.

2.4. Podsumowanie.

W następnych dwóch rozdziałach pracy przedstawione zostaną założenia trzech podstawowych polityk sterowania dostępem oraz skojarzonych z nimi mechanizmów, tzn.

- polityki uznaniowego sterowania dostępem (Discretionary Access Control -DAC),
- polityki narzuconego sterowania dostępem (Mandatory Access Control - MAC),
- polityki sterowania dostępem w oparciu o role (Role Based Access Control - RBAC).

Zagadnienie ochrony bezpieczeństwa systemu informatycznego jest zadaniem wielce złożonym, dlatego jest ono zazwyczaj dekomponowane na kilka niezależnych domen takich jak audyt, sterowanie dostępem, dostępność, itp. Elementem, który odgrywa znaczącą rolę w całym procesie ochrony, a zwłaszcza we współczesnych systemach charakteryzujących się rozległością przestrzenną i wielodostępnością, jest logiczne sterowanie dostępem. Podstawą skutecznego sterowania dostępem jest dobrze zdefiniowana polityka oraz mechanizmy pozwalające na realizację tejże polityki. Proces tworzenia polityk i mechanizmów sterowania dostępem jest wciąż przedmiotem różnorodnych badań, mających na celu jego usprawnienie.

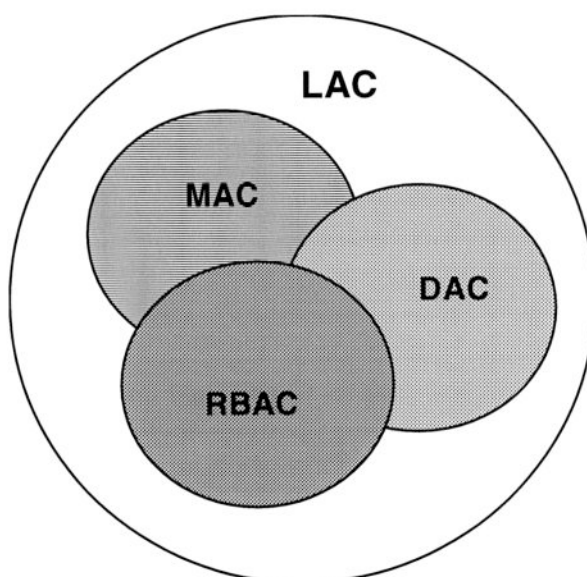
W celu uproszczenia dalszych rozważań, przyjęto założenie o poprawności funkcjonowania wszystkich niezbędnych elementów systemu bezpieczeństwa znajdujących się powyżej poziomu polityk sterowania dostępem.

3. Przegląd podstawowych polityk sterowania dostępem.

W rozdziale tym scharakteryzowane zostaną najważniejsze własności dwóch polityk sterowania dostępem, a mianowicie: polityki uznaniowego sterowania dostępem (ang. Discretionary Access Control - DAC) i polityki narzuconego sterowania dostępem (ang. Mandatory Access Control - MAC).

Polityka sterowania dostępem poprzez role (ang. Role Based Access Control - RBAC), która stanowi główny przedmiot badań tej pracy, zostanie szerzej przedstawiona w następnym rozdziale.

Opisanych w tym i w następnym rozdziale polityki sterowania dostępem nie należy traktować jako rozwiązań konkurencyjnych czy też wzajemnie się wykluczających. Mogą one stanowić samodzielną podstawę dla tworzenia mechanizmów sterowania dostępem w systemie, jak i współdziałać ze sobą w celu zapewnienia większej lub bardziej wszechstronnej i elastycznej ochrony. Ilustruje to praca R.Sandhu, który przedstawił zależność pomiędzy trzema politykami sterowania dostępem w postaci trzech nierozłącznych zbiorów (rys.3.1.) [SS94]. Poszczególne podzbiory ilustrują możliwe do zaimplementowania kombinacje polityk. Nie oznacza to jednak, że łączenie wielu polityk w jednym systemie jest prostym zadaniem. Na skutek błędów lub przeoczenia ze strony projektanta systemu, czy też jego administratora, może dojść do sytuacji, w której system będzie wypracowywał sprzeczne ze sobą decyzje co do możliwości wykonywania przez podmiot danej operacji. W przypadku łączenia w jednym systemie wielu polityk, należy zatem szczególną uwagę poświęcić problemowi badania spójności i zarządzania zależnościami pomiędzy politykami [SS94].



rys.3.1. Zależności pomiędzy politykami logicznego sterowania dostępem.

Niezbędnym elementem logicznego sterowania dostępem obok polityki sterowania są mechanizmy sterowania dostępem. Mechanizm dostarcza technicznych środków umożliwiających implementację przyjętej polityki bezpieczeństwa. Wśród tych środków znajdować powinien się model opisujący warunki dostępu i weryfikacji bieżącego stanu systemu oraz pojawiających się żądań dostępu.

Podstawowym modelem na podstawie którego powstało wiele różnorodnych mechanizmów sterowania dostępem jest macierz dostępu (ang. access matrix). Pierwszą w pełni formalną wersję macierzy dostępu na potrzeby modelowania sterowania dostępem opracował Lampson we współpracy z COSINE Task Force on Operating Systems [Lam74]. Model ten został następnie ulepszony przez Grahama i Deninga [Den74], a następnie Harrison, Ruzzo i Ulman przedstawili wersję macierzy dostępu opracowaną na potrzeby dowodzenia własności mechanizmów ochrony [Den74],[Den76],[Den86].

Podstawowymi pojęciami dla modeli związanych z macierzą dostępu są stany i przejścia pomiędzy stanami. Stan systemu ochrony jest opisywany macierzą, a przejścia stanów reprezentują polecenia systemu.

Model ochrony oparty o macierz dostępu można zdefiniować najprościej w następujący sposób:

Niech stan systemu będzie reprezentowany poprzez trójkę (S,O,M), gdzie:

- S - zbiór podmiotów,
- O - zbiór obiektów,
- M - macierz dostępu (dostęp może być jedynie typu - odczyt/zapis),

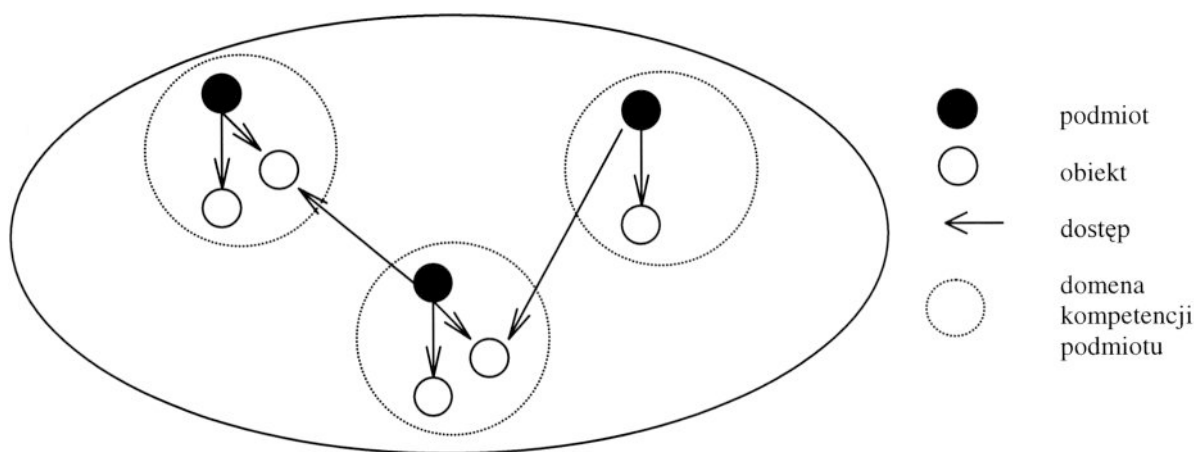
Wierszom macierzy odpowiadają podmioty, kolumnom zaś obiekty. Elementy macierzy $M[s,o]$ zawierają listę praw dostępu podmiotu s do obiektu o . Mogą one zawierać jedynie prawa elementarnego typu i jest ich skończona ilość.

Podmiot\Obiekt	Port_Drukarki	Pliki_dane_1	Port_FTP
Użytkownik_1	<i>zapis</i>	<i>zapis,odczyt</i>	<i>odczyt</i>
Użytkownik_2	<i>zapis</i>	<i>odczyt</i>	<i>zapis,odczyt</i>
Gość		<i>odczyt</i>	<i>odczyt</i>
Administrator	<i>zapis</i>	<i>zapis,odczyt</i>	<i>zapis,odczyt</i>

rys.3.2. Przykładowa macierz dostępu

3.1. Uznaniowe sterowanie dostępem - DAC.

Uznaniowe sterowanie dostępem (DAC) jest polityką, która umożliwia ograniczanie dostępu do obiektów systemu w oparciu o tożsamość podmiotów i ich przynależność do wyróżnionych w systemie grup. Sterowanie jest uznaniowe w tym sensie, że podmiot posiadający prawo sterowania, jest w stanie przekazać swoje uprawnienia (być może, że w pośredni sposób), dowolnemu innemu podmiotowi [NCSC87a].



rys. 3.3. Idea sterowania uznaniowego.

Do najważniejszych zagadnień związanych z polityką DAC należy zaliczyć sposób zarządzania prawem do sterowania. Prawo do sterowania jest wyróżnionym obok podstawowych typów dostępu takich jak zapis, odczyt, itp., prawem, które podmiot może posiadać w stosunku do obiektu. Umożliwia ono podmiotowi wyznaczanie polityki dostępu do obiektu względem którego podmiot to prawo posiada. Podmiot dysponujący prawem do sterowania obiektem *X*, może zezwolić na dostęp dowolnego typu, dowolnemu podmiotowi (grupie podmiotów) bądź też zakazać takiego dostępu. W szczególnym przypadku, właściciel prawa sterowania, może przekazywać innym podmiotom to prawo. Możliwość taka jest zgodna z ogólną filozofią DAC, która zakłada swobodę kształtowania polityki dostępu do zasobów przez szeroką grupę podmiotów. Niemniej jednak nadmierny liberalizm przy gospodarowaniu prawem sterowania jest niewskazany ze względu na możliwość utraty sterowania nad dostępem, a w efekcie i bezpieczeństwa systemu. Dzieje się tak przeważnie w przypadku wystąpienia konfliktu kilku koncepcji ochrony, reprezentowanych przez różne podmioty. W związku z powyższym, przyjęty sposób zarządzanie prawem sterowania wyznacza ogólne zasady zarządzania dostępem w systemie

Wyróżnia się cztery główne metody zarządzania prawem sterowania w systemach z polityką DAC.

a) *Podejście hierarchiczne.*

Hierarchiczny model zarządzania sterowaniem dostępem jest oparty na naturalnej, istniejącej w danej organizacji hierarchii odzwierciedlającej możliwości i kompetencje podmiotów. Główny administrator systemu tworzy ze zbioru wszystkich obiektów podzbiory (grupy), którym przypisuje odpowiedzialne za nie podmioty, posiadające względem tych

obiektów pełnię praw z prawem sterowania włącznie. Podmioty te z kolei wydzielają z przyznanych im zasobów rozłączne podzbiory obiektów określając, który z podmiotów będzie od tej pory odpowiedzialny za dany podzbiór i przyznają im potrzebne prawa (mogą przyznać również prawo sterowania). Analogiczny proces postępuje, aż do ustalonego poziomu elementarnych zasobów oraz podmiotów dysponującymi jedynie prawami dostępu, bez możliwości sterowania jakimkolwiek z zasobów.

Zaletą tego podejścia jest to, iż prawo sterowania przysługuje jedynie wydzielonym podmiotom, które biorą odpowiedzialność za stan systemu. Odpowiedzialność ta oznacza zarówno zaufanie w stosunku do nich, jak i możliwość łatwego zidentyfikowania nadużyć, a w związku z tym możliwość wyciągnięcia odpowiednich konsekwencji [MS93] [MS94].

b) Podejście własnościowe.

Zarządzanie sterowaniem dostępem w oparciu o pojęcie własności podlega na domyślnym przekazaniu prawa do sterowania właścicielowi obiektu. Właścicielem obiektu jest zazwyczaj jego twórca, bądź wyróżniony jako nadrzędny podmiot systemu. W związku z tym jedynie właściciel obiektu posiada pełnię praw i może je w sposób swobodny udzielać innym podmiotom. Nie może on jednak, w tym przypadku przekazywać posiadanego prawa sterowania. Dlatego też, model własnościowy polityki DAC może być postrzegany jako szczególny przypadek podejścia hierarchicznego z jednostopniową hierarchią. Model ten cechuje łatwość implementacji, gdyż wystarczy aby podczas tworzenia obiektu zostało automatycznie przyznane podmiotowi tworzącemu również i prawo sterowania tym obiektem. Innym ujęciem tej metody jest implementacja bez wyodrębniania oddzielnego prawa sterowania, a w zamian tego automatycznego przyznawania możliwości manipulowania na prawach dostępu właścicielom obiektów [NCSC87a], [BN89].

Niedogodnością takiego podejścia jest konieczność pośredniczenia właściciela obiektu we wszelkich operacjach modyfikujących prawa dostępu do obiektu.

Prawo sterowania powiązane z atrybutem własności jest cechą charakterystyczną dla wszystkich systemów opartych o architekturę systemu Unix.

c) Podejście „Laissez-Faire”

Jest to najbardziej liberalny model sterowania dostępem do obiektów oparty na polityce DAC. W systemie realizującym ten sposób przekazywania uprawnień nie ma wyróżnionych właścicieli obiektów. Każdy podmiot posiadający prawo sterowania może je w sposób dowolny przekazać jakiegokolwiek innemu podmiotowi, który może następnie uczynić to samo. Model ten nie jest w związku z tym w stanie zapobiegać niekontrolowanemu rozprzestrzenianiu się uprawnień. [NCSC87a], [Lun88].

W celu wyeliminowania tej wady podejście „Laissez-Faire” przy jednoczesnym dopuszczeniu do znacznej swobody gospodarowania prawami sterowania, powstały modele, które np. pozwalają tylko jednokrotnie (dwukrotnie, trzykrotnie) przekazać prawo sterowania danym obiektem, a następnie możliwość ta jest blokowana [BC92].

d) Podejście centralistyczne.

Zgodnie z nazwą tego sposobu sterowania dostępem w systemie zostaje wyróżniony jeden podmiot, który jest nadrzędny w stosunku do pozostałych. Posiada on wyłączne prawo do nadawania i odbierania praw dostępu do wszystkich istniejących obiektów. Nie może on jednak przekazywać swojego prawa sterowania żadnemu innemu podmiotowi. Wady tego podejścia są takie same jak w przypadku wszystkich innych systemów ze scentralizowanym zarządzaniem, tzn. mała skalowalność i wąskie gardło w systemie zarządzania w postaci jednego podmiotu nadzorującego [NCSC87a].

3.1.1. Mechanizmy polityki DAC.

Dla poprawnego funkcjonowania systemu sterowania dostępem istotne są również mechanizmy implementujące wybraną politykę. Mechanizmy wspierające politykę DAC w większości są modyfikacją mechanizmu sterowania dostępem w oparciu o macierz dostępu (access matrix). Poniżej przedstawiono pięć podstawowych mechanizmów wspierających politykę DAC, które stanowią próbę zwiększenia efektywności wyjściowego mechanizmu macierzy dostępu, poprzez uwzględnienie specyfiki konkretnego systemu [NCSC87a].

a) Możliwości (ang. Capabilities)

System oparty na tym mechanizmie steruje dostępem do obiektów, sprawdzając czy podmiot zgłaszający żądanie dostępu, posiada odpowiedni znacznik. Znaczniki określają obiekt oraz rodzaj dozwolonego dostępu do tego obiektu. Podmiot może wykonywać jedynie operacje, których typ jest określony w posiadanych przez niego znacznikach i tylko na obiektach, których znaczniki te dotyczą. Podmioty mogą przekazywać sobie znaczniki, lecz nie mogą zmieniać ich zawartości, ani tworzyć nowych znaczników.

b) Profile (ang. Profiles)

Mechanizm ten tworzy listy obiektów stowarzyszonych z każdym z podmiotów. Jest to sposób bardzo nieefektywny pod względem zapotrzebowania na pamięć operacyjną i na czas potrzebny na obsługę żądań dostępu. Duże zapotrzebowanie na pamięć jest wynikiem tego, że ilość profili jest równa ilości podmiotów, zaś ich wielkość jest wprost proporcjonalna do ilości obiektów. Zapotrzebowanie na czas wynika zaś z tego, że ustalenie praw dostępu do obiektu wiąże się z koniecznością przeglądania wszystkich profili. Podobnie dodawanie i kasowanie obiektów wymaga przeglądnięcia wszystkich i uaktualnienia dużej liczby profili.

c) Listy sterowania dostępem (ang. Access Control List - ACL)

Listy sterowania dostępem implementują macierz dostępu tworząc dla każdego obiektu listę podmiotów i dostępnych dla każdego z nich praw dostępu. Elementami listy mogą być zarówno pojedyncze podmioty, grupy, bądź grupy podmiotów wyznaczone poprzez nazwę zawierającą metaznaki (takie jak '*' i '?'). Listy sterowania dostępem stanowią dogodny sposób zarządzania sterowaniem dostępem dla obiektów tworzących strukturę drzewiastą.

Poprzez zastosowanie dziedziczenia uprawnień z obiektów nadrzędnych, można zredukować konieczność ustalania praw dla każdego podmiotu z osobna.

Innym użytecznym rozszerzeniem list sterowania dostępem jest możliwość zastosowania tzw. *Named ACL*. Jeżeli w systemie do nowo tworzonych obiektów przypisywany jest przeważnie stały zbiór podmiotów z ustalonymi prawami dostępu, możliwe jest stworzenie szablonu - *Named ACL*, który można odpowiednią ilość razy powielić, w celu aktualizacji zbioru praw dostępu dla nowotworzonych obiektów.

d) Bity ochronne (ang. Protection bits)

Koncepcja bitów ochronnych polega na powiązaniu z każdym z obiektów określonej ilości bitów informacyjnych. Bity te umożliwiają zapis praw dostępu do obiektu dla podmiotów z ze zbioru kilku kategorii użytkowników takich jak: właściciel, grupa, pozostali użytkownicy. Przykładem implementacji tego mechanizmu są systemy oparte na architekturze Unix. Przyjmuje się w tych systemach, m.in. że twórca obiektu staje się jednocześnie jego właścicielem i jako taki posiada pełnię praw względem obiektu wraz z możliwością udzielania ich innym podmiotom.

e) Hasła (ang. Passwords)

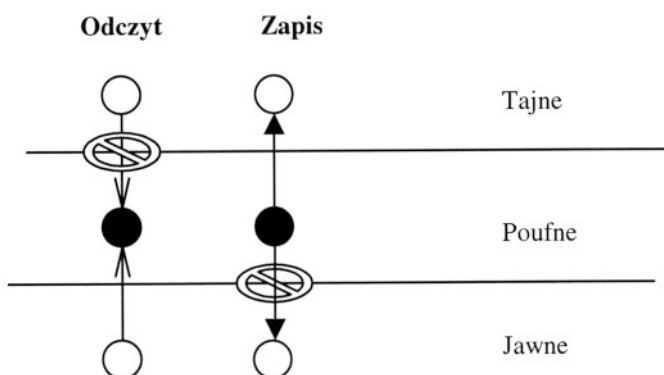
Metoda uznaniowego sterowania dostępem oparta o mechanizm haseł polega na ochronie zasobów poprzez uzależnienie możliwości dostępu do nich od znajomości przez podmiot hasła dostępu. Podmiot zamierzający wykonać określone operacje na obiekcie, w celu uzyskania zezwolenia na ich wykonanie, musi każdorazowo wykazać się znajomością odpowiedniego hasła. Możliwe są do przyjęcia dwie koncepcje ochrony w tym mechanizmie: indywidualna i grupowa. W pierwszym przypadku każdy podmiot dysponuje indywidualnie przyznanymi uprawnieniami i własnym zbiorem haseł. W przypadku grupowego sterowania, dana grupa podmiotów posiada jednakowe uprawnienia i jednakowe hasło.

3.2. Narzucone sterowanie dostępem - MAC.

Narzucone sterowanie dostępem (ang. Mandatory Access Control - MAC) jest stosowane w systemach ze ścisłą klasyfikacją podmiotów i obiektów. Każdy obiekt w obrębie działania polityki MAC posiadać powinien przyznaną mu klasę ochrony. Im większe znaczenie dla systemu ma obiekt i przechowywana w nim informacja, tym wyższa klasa ochrony powinna być z nim skojarzona. W analogiczny sposób powinny zostać sklasyfikowane podmioty. Im bardziej godny zaufania podmiot, im przysługują mu większe możliwości w systemie, tym przypisana mu klasa ochrony powinna być wyższa [Lee88], [Lea90], [Kol00b], [LS90], [San93], [Sut86].

W najprostszym przypadku klasy ochrony tworzą prostą uporządkowaną strukturę hierarchiczną wraz z ustaloną relacją porządkującą. Można ją odnieść do klasyfikacji dokumentów w instytucjach rządowych, czy też militarnych gdzie zazwyczaj wyróżnia się następujące klasy tajności dokumentów: ściśle tajne (ST), tajne (T), poufne (P) i jawne (J). Wyżej wymienione poziomy możemy uszeregować przy użyciu relacji '>' w następujący sposób: ST>T>P>J. Oznacza to, że poziom ST jest na szczycie hierarchii klas ochrony i charakteryzuje się większą ochroną, niż pozostałe klasy względem niego podrzędne.

Podmiot może uzyskać dostęp do obiektu w zależności od relacji, która zachodzi pomiędzy poziomem jego klasy ochrony, a poziomem obiektu do którego chce uzyskać dostęp. Dla operacji odczytu obowiązuje zasada *czytaj niżej* (ang. *read down*), która mówi, iż podmiot może czytać z obiektów dla których spełniona jest relacja, iż jego poziom klasy ochrony dominuje nad poziomem klasy ochrony obiektu. Dla operacji zapisu zdefiniowana jest analogiczna zasada *pisz wyżej* (ang. *write up*), która mówi, że podmiot może dokonywać zapisu do obiektów tylko wtedy, gdy jego poziom klasy ochrony jest mniejszy od poziomem klasy ochrony obiektu względem którego chce tę operację wykonać.



rys.3.4. Sterowanie narzucone. Jednokierunkowy przepływ informacji w systemie

Tak sformułowane warunki odnośnie zapisu i odczytu wyznaczają zasadę jednokierunkowego przepływu informacji (ang. One Direction Information Flow) [Lea90], [Den82]. Zgodnie z nią, informacja w systemie informatycznym może przepływać tylko w jednym kierunku, tzn. od obiektów i podmiotów o niższej klasie ochrony, do obiektów i podmiotów wyżej sklasyfikowanych. Ma to na celu zabezpieczenie przed nieuprawnionym wykorzystywaniem informacji o dużym znaczeniu dla bezpieczeństwa systemu czy też funkcjonowania organizacji, przez podmioty o niskim poziomie zaufania.

Powyższa zasada niesie ze sobą jednak pewne niedogodności, a mianowicie:

- podmiot pisząc do obiektu o wyższej klasie ochrony niż jego własny, nie jest w stanie kontrolować tej operacji, tzn. sprawdzić poprzez odczyt co i jak zostało w rzeczywistości zapisane,
- podmiot o wysokiej klasie ochrony nie może pisać do obiektów o niższym poziomie, w szczególności nie możliwe jest wydawanie przez „przełożonych” dyspozycji ich „podwładnym”.

W celu wyeliminowania tych wad podstawowego modelu sterowania narzuconego, w wielu przypadkach modyfikuje się warunki *pisz wyżej i czytaj niżej*. Umożliwia to uzyskanie pożądanego efektu sterowania przepływem informacji z jednej strony, ale i dostosowania do rzeczywistych potrzeb systemu i organizacji z drugiej. Modyfikacja taka polegać może np. na wprowadzeniu do systemu oprócz hierarchicznych poziomów bezpieczeństwa, również niehierarchicznych kategorii. W ten sposób kategoria skojarzona z podmiotem wyznacza domenę jego działalności. Kategoria powiązana z obiektem, określa zaś obszar w którym mogą rozchodzić się informacje w nim zawarte [Cup93], [Lea90].

3.2.1. Mechanizmy MAC.

Przykładem formalnego mechanizmu polityki MAC jest model opracowany przez Bell i LaPadula oparty na wcześniejszych badaniach Lampsona, Harrisona i Ruzzo [Den86]. Model ten pozwala na formalną analizę i dowodzenie własności systemu działającego zgodnie z przyjętymi przez autorów założeniami. Założenia te są jak następuje:

- System opisany jest przez trójkę (S,M,O), gdzie:
 - S - zbiór podmiotów,
 - O - zbiór obiektów,
 - M - macierz dostępu (dostęp może być jedynie typu - odczyt/zapis),
- Jest określona krata poziomów zaufania dla podmiotów i obiektów,
- $F: S \cup O \rightarrow L$ - funkcja, której argumentem jest obiekt lub podmiot, rezultat zaś jest poziomem zaufania danego elementu,
- $V=(F,M)$ - zbiór stanów, w którym wyróżnia się jeden ze stanów v_0 jako stan początkowy,
- R - zbiór żądań systemu (akcji),
- $T(V \times R) \rightarrow V$ - funkcja obrazująca przejścia systemu w kolejne stany po realizacji żądań.

Dla systemu spełniającego powyższe założenia Bell i La Padulla podali definicję bezpieczeństwa systemu.

1. Stan (F,M) jest bezpieczny ze względu na operację odczytu wtedy i tylko wtedy gdy $\forall s \in S \forall o \in O \text{ read} \in M[s,o] \rightarrow F(s) \geq F(o)$,
2. Stan (F,M) jest bezpieczny ze względu na operację zapisu wtedy i tylko wtedy gdy $\forall s \in S \forall o \in O \text{ write} \in M[s,o] \rightarrow F(o) \geq F(s)$,
3. Stan jest bezpieczny wtedy i tylko wtedy gdy jeżeli jest jednocześnie bezpieczny ze względu na operację zapisu i odczytu,
4. System (v_0, R, T) jest bezpieczny wtedy i tylko wtedy gdy v_0 jest stanem bezpiecznym i wszystkie stany osiągalne z v_0 poprzez wykonanie operacji z R są bezpieczne.

3.3. Podsumowanie.

W rozdziale tym scharakteryzowano najważniejsze założenia związane z polityką uznaniowego sterowania dostępem (DAC) oraz z polityką narzuconego sterowania dostępem (MAC).

Dla polityki DAC przedstawiono cztery główne metody zarządzania prawem sterowania, tzn.: podejście hierarchiczne, własnościowe, „laissez-faire” oraz centralistyczne. Następnie omówione zostały najważniejsze mechanizmy wspierające tę politykę, tzn.: możliwości, profile, listy sterowania dostępem, bity ochronne i hasła.

Dla polityki MAC omówiono ideę sterowania dostępem w oparciu o istniejącą w systemie klasyfikację podmiotów i obiektów oraz przedstawiono podstawowy mechanizm tej polityki opracowany przez Bell i LaPadula wspierający tę politykę.

4. Polityka i mechanizmy RBAC.

W pierwszej części tego rozdziału zostaną scharakteryzowane najważniejsze cechy polityki sterowania dostępem poprzez role (RBAC). Następnie zostanie przedstawiona krytyczna analiza najważniejszych spośród istniejących implementacji tej polityki. Analiza przeprowadzona w tej części pracy posłuży do precyzyjnego zidentyfikowania problemów związanych z implementacją mechanizmów wspierających politykę sterowania dostępem poprzez role. Spostrzeżenia te wykorzystane zostaną w dalszej części pracy, gdzie przedstawiona zostanie propozycja nowego modelu mechanizmu, pozwalającego zmniejszyć lub zlikwidować całkowicie istniejące niedogodności, a przez to zwiększyć efektywność sterowania dostępem poprzez role.

4.1. Postulaty sterowania dostępem poprzez role.

Polityki sterowania dostępem MAC i DAC kształtowały się przez wiele lat. Od chwili powstania systemów wielodostępnych istniała konieczność ochrony ich zasobów przed nieuprawnionym dostępem. Właśnie wtedy powstawały pierwsze opracowania teoretyczne i implementacje dwóch wcześniej omawianych polityk. Ze względu na swoje zalety, jak i często z powodu braku alternatywy, wiele współczesnych systemów opiera sterowanie dostępem właśnie na jednej, bądź na połączeniu obu tych polityk. Badania nad prezentowaną w tym rozdziale polityką sterowania dostępem poprzez role, zostały zintensyfikowane dopiero na początku lat dziewięćdziesiątych. Z tego też okresu wywodzi się większość prac teoretycznych tej polityki jak i jej pierwsze próby implementacji [FCK95], [FK95], [FK92], [Gui95], [Kol00a], [BCFGK97], [Bar95a], [ISO95], [Loi94].

U podstaw RBAC leży spojrzenie na system informatyczny jako integralny element określonej organizacji, którego głównym zadaniem jest wspieranie celu funkcjonowania tejże organizacji. Podmioty działające w systemie reprezentują bądź pracowników organizacji, bądź też jakiś fragment jej działalności, a w związku z tym może być z nimi skojarzony również zbiór zadań, które wykonują i za których wykonanie ponoszą odpowiedzialność. Aktywność podmiotu w systemie informatycznym należy traktować zatem jako jeden z kolejnych elementów na drodze realizacji ogólnego celu funkcjonowania organizacji.

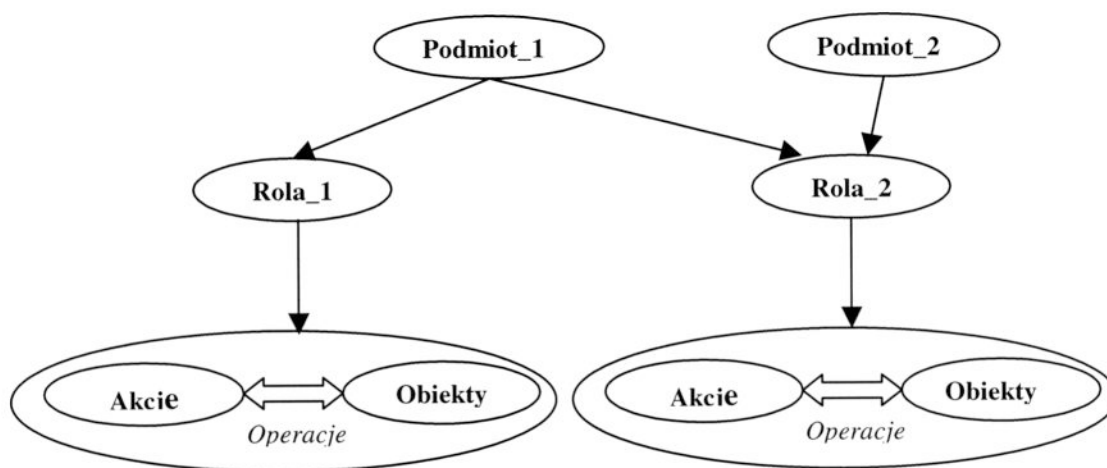
Polityka RBAC steruje dostępem podmiotów do zasobów w oparciu o wyróżnione w organizacji i systemie role. Rola jest zdefiniowana w następujący sposób:

Rola - jest zbiorem uprawnień oraz obowiązków przysługującym przypisanym do niej podmiotom

Koncepcja ról pojawiła się w teorii organizacji o wiele wcześniej niż powstały systemy informatyczne [OL96], [SS94]. Role były wykorzystywane w tej teorii w celu definiowania bezpiecznego i efektywnego procesu zarządzania organizacją. Dlatego też, właściwie zaimplementowana polityka RBAC może być elementem wspierającym nie tylko bezpieczeństwo systemu informatycznego danej organizacji, ale i przyczynić się do usprawnienia zarządzania całą organizacją [FB97].

Polityka RBAC wymaga określenia trzech podstawowych dla niej relacji:

- podmiot - rola,
- uprawnienia - rola,
- rola - rola.



rys.4.1. Idea polityki RBAC

Przypisanie podmiotów do ról odbywa się na zasadzie identyfikacji zadań jakie pełnią one w systemie, a konkretnie zakresu kompetencji i obowiązków na nich spoczywających. Powinno być możliwe płynne przypisywanie i odwoływanie przypisania podmiotów do ról w sposób nie zakłócający funkcjonowania innych elementów systemu. Role ułatwiają zarządzanie możliwościami podmiotów, gdyż poprzez pojedynczą operację przypisania do roli, podmiot uzyskuje wszystkie potrzebne prawa.

Stosunkowo prosta jest również operacja uaktualniająca prawa podmiotów, kiedy zmienia się charakter roli, którą odgrywają w systemie. Zmieniany jest wtedy profil uprawnień stowarzyszonych z rolą, uzyskując jednocześnie pewność, że wszystkie podmioty przypisane do tej roli będą od tego momentu dysponować nowo wyznaczonym zbiorem uprawnień.

Przy dobrze zdefiniowanych rolach, polityka RBAC wspiera jedną z podstawowych zasad bezpieczeństwa systemów informatycznych, tzn. zasadę najmniejszego uprzywilejowania. Możliwe jest to, wtedy gdy role są zbiorem minimalnych, koniecznych i wystarczających uprawnień do realizacji zadań wykonywanych przez przypisanie do nich podmioty. Automatycznie zmniejszone zostaje również prawdopodobieństwo pomyłki podczas tworzenia profilu uprawnień dla nowego podmiotu oraz ułatwione późniejsze zarządzanie

uprawnieniami, poprzez niedopuszczenie aby przy zmianie zadania wykonywanego przez podmiot zostało mu przyznane zbyt mało praw, albo aby wciąż dysponował prawami z poprzedniego zadania, które obecnie nie są dla niego konieczne. Proces przypisywania praw do roli jest elementem krytycznym polityki ze względu na bezpieczeństwo systemu. Polityka RBAC umożliwia sterowanie dostępem na poziomie elementarnych praw dostępu (write, read, itp.), ale pozwala również przenieść sterowanie na dowolnie wysokim poziomie abstrakcji, tzn. poprzez odnoszenie się do możliwości wykonywania przez podmioty *transakcji*. Termin *transakcje* należy rozumieć jako nierozzerwany ciąg elementarnych akcji dostępu. Wyróżnione transakcje powinny być zgodne z wymogami bezpieczeństwa organizacji, których nie da się wyrazić przy pomocy sterowania pojedynczymi akcjami dostępu. Przykładem wykorzystania *transakcji* do opisu akcji dostępu do zasobów systemu może być sytuacja kasjera w banku i nadzorcy. Kasjer może dokonywać operacji na rachunkach, lecz nie może ich korygować, podczas gdy nadzorca posiada jedynie prawo modyfikowania istniejących operacji. Opisanie tej zależności na poziomie podstawowych praw dostępu byłoby bardzo trudne, zaś przy pomocy opisu na poziomie operacji, np. *korekta, operacja_na_rachunku*, uzyskujemy łatwość rozróżnienia obu typów dostępu.

Bardzo ważnym dla jakości sterowania jest również sposób zarządzania relacjami pomiędzy rolami. Aby być w zgodzie z stanem rzeczywistym danej organizacji wykorzystującej system informatyczny, polityka RBAC musi uwzględniać możliwość istnienia zależności pomiędzy rolami np. priorytetów ról, ról wzajemnie wykluczających się, ról wzajemnie wymagających się, działania jednego podmiotu w kilku rolach jednocześnie, możliwości tymczasowego delegowania ról, itp. Podstawowym elementem polityki sterowania dostępem poprzez role jest zazwyczaj hierarchia ról, odzwierciedlająca naturalną strukturę organizacji. Na podstawie tej hierarchii powinno prowadzić się dalsze rozważania tzn. badać odpowiedniość systemu ról i rzeczywistych wymagań organizacji, likwidować konflikty pomiędzy rolami, identyfikować istniejące zależności.

Za jedną z głównych zalet tej polityki sterowania dostępem można uznać to, że umożliwia ona zarządzanie dostępem na wysokim poziomie abstrakcji typowym dla organizacji komercyjnych. Powoduje to, że proces zarządzania dostępem nie wymaga dłuższej oddzielnych procedur dostosowujących wymagania organizacji do możliwości systemu informatycznego. Jest to możliwe, gdyż RBAC opiera się na wyróżnionych w organizacji rolach, które określone są poprzez zbiór uprawnień oraz obowiązków i skojarzonych z nimi podmiotów. W działającym systemie, większość operacji związanych z administracją systemem sterowania dostępem ogranicza się do przypisywania podmiotów do ról, bądź do ich usuwania. Jest to znacząca poprawa w stosunku do konieczności ciągłego manipulowania na poziomie elementarnych praw dostępu w systemach opartych o mechanizmy *ACL*, *macierz dostępu*, itp. Zaletą tej polityki jest również możliwość istnienia zarówno jednego jak i dowolnie wielu podmiotów nadrzędnych, które to mogą posiadać w zależności od potrzeb pełnię praw nadzorcy systemu, bądź też jedynie dowolnie ustalony podzbiór tych praw taki jak np. prawo do tworzenia hierarchii ról, definiowania ról, przypisywania podmiotów do ról, itd.

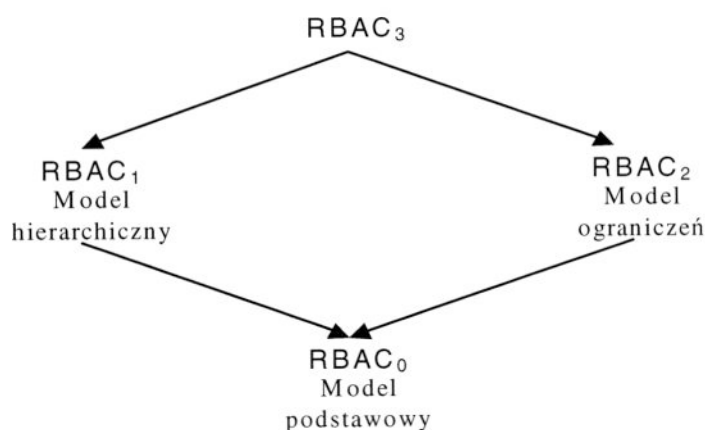
4.2. Wybrane mechanizmy polityki RBAC.

4.2.1. Model RBAC'96 i ARBAC'97.

Wypracowany w połowie lat dziewięćdziesiątych w National Institute of Standards and Technology (NIST) model polityki RBAC składa się z dwóch zasadniczych części: modelu RBAC'96 - opisującego różnorodne aspekty funkcjonalne sterowania dostępem poprzez role oraz modelu ARBAC'97 - będącego próbą opisu problematyki związanej z zarządzaniem rolami [San97], [SB97], [SBCG97], [SBM99].

4.2.1.1. RBAC'96

RBAC'96 składa się z rodziny czterech modeli. Modele te stanowią próbę formalnej definicji poszczególnych aspektów sterowania dostępem poprzez role na potrzeby mechanizmów realizujących tę politykę. Model $RBAC_0$ jest podstawowy dla całej rodziny zaś pozostałe modele stanowią jego rozszerzenia odpowiednio o hierarchię ról ($RBAC_1$) i o ograniczenia ($RBAC_2$). Model $RBAC_3$ pozwala na wykorzystanie zarówno hierarchii jak i ograniczeń.



rys.4.2. Zależności pomiędzy elementami modelu RBAC'96

a) $RBAC_0$

Podstawą opisu polityki RBAC jest $RBAC_0$, który zdefiniowany jest przez następujące elementy:

U, R, P, S	- zbiory odpowiednio - użytkowników (podmiotów), ról, praw dostępu i sesji,
$PA \subseteq R \times P$	- relacja przyporządkowania praw dostępu do poszczególnych ról,
$UA \subseteq U \times R$	- relacja przyporządkowująca użytkowników do ról,
user: $S \rightarrow U$	- funkcja przypisująca każdej sesji s_i jednego użytkownika $user(s_i)$,
roles: $S \rightarrow 2^R$	- funkcja mapująca każdą sesję s_i na zbiór ról $\{roles(s_i) \subseteq \{r (user(s_i), r) \in UA\}$ i sesja s_i ma następujące praw dostępu $\cup_{r \in ROLES(s_i)} \{p (p, r) \in PA\}$

Prawa dostępu są traktowane w $RBAC_0$ jedynie jako prosty zbiór symboli pozbawionych interpretacji. Według autorów, ma to zapewnić większą elastyczność modelu, albowiem natura prawa dostępu jest wysoce zależna od konkretnej implementacji i systemu.

Zgodnie z przyjętą definicją, jedynie te operacje są dozwolone, do których uprawnienie zostało w sposób bezpośredni przyznane (tzn. elementom PA dla danej roli). Wszystkie inne operacje należy traktować jako zabronione.

Zbiór praw dostępu odnosi się jedynie do obiektów systemu nie związanych z zarządzaniem sterowaniem dostępem. Prawa pozwalające na modyfikację polityki sterowania dostępem, czyli zbiorów U, R, P, S, PA, UA są rozpatrywane oddzielnie, w części charakteryzującej zarządzanie RBAC (punkt 4.2.1.2. odnośnie modelu ARBAC'97).

b) $RBAC_1$

$RBAC_1$ zawiera w sobie wszystkie elementy $RBAC_0$ i dodatkowo wprowadza hierarchię ról (RH), która jest jednym z elementów stanowiących o atrakcyjności polityki RBAC [FK92], [HDT95], [NO95]. Hierarchia jest środkiem modulującym strukturę zdefiniowanego zbioru ról i umożliwiającym odzwierciedlenie istniejących linii podziałów odpowiedzialności i kompetencji z poziomu organizacji.

Przypuśćmy, że w organizacji mamy zdefiniowane dwie następujące role: *Kierownik_działu_techicznego* i *Pracownik_działu_techicznego*. Każdy pracownik ma określony zbiór uprawnień, które są również dostępne dla jego kierownika oraz kierownik posiada również pewne dodatkowe uprawnienia, które nie są z kolei dostępne dla jego podwładnych. Właśnie umożliwieniu reprezentowania tego typu sytuacji służy wprowadzenie hierarchii ról. Gdy mamy możliwość określenia hierarchicznych zależności pomiędzy rolami, to po pierwsze unikamy nadmiarowości przy definiowaniu ról (tzn. możemy określić zbiór uprawnień kierownika jako zbiór uprawnień pracownika plus dodatkowe uprawnienia), a po

drugie uzyskujemy bardziej przejrzysty zapis odzwierciedlający powiązania w organizacji i systemie co może usprawnić proces administrowania.

Z matematycznego punktu widzenia hierarchia jest relacją częściowo uporządkowaną, tzn. jest zwrotna, przechodnia i nie jest symetryczna.

- | | | |
|-----------------------------------|---|--|
| $U, R, P, S, PA, UA, \text{user}$ | - | tak jak w $RBAC_0$ |
| $RH \subseteq R \times R$ | - | relacja częściowo uporządkowana na zbiorze R , hierarchia ról lub relacja dominacji zapisywana jest w postaci graficznego znaku ' $\geq$ ' |
| $\text{roles}: S \rightarrow 2^R$ | - | zmodyfikowana z $RBAC_0$ funkcja - $\text{roles}(s_i) \subseteq \{r \mid \exists r' \geq r [(user(s_i), r') \in UA]\}$ sesja posiada prawa dostępu $\bigcup_{r \in \text{roles}(s_i)} \{p \mid \exists r'' \leq r [(p, r'') \in PA]\}$ |

Przykład 4.1. HIERARCHIA RÓL.

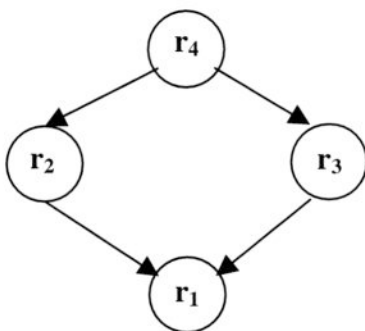
Przyjmijmy, że mamy system z następująco zdefiniowanymi zbiorami:

- | | |
|--------------|------------------------------------|
| role: | $R = \{r_1, r_2, r_3, r_4\},$ |
| prawa: | $P = \{p_1, p_2, p_3, p_4, p_5\},$ |
| użytkownicy: | $U = \{u_1, u_2, u_3, u_4\},$ |

oraz relacjami:

- | | |
|-----|--|
| PA: | $\{ \langle r_1, p_1 \rangle, \langle r_2, p_2 \rangle, \langle r_2, p_3 \rangle, \langle r_3, p_4 \rangle, \langle r_4, p_5 \rangle \}$ |
| UA: | $\{ \langle u_1, r_1 \rangle, \langle u_2, r_2 \rangle, \langle u_3, r_3 \rangle, \langle u_4, r_4 \rangle \}$ |
| RH | $\{ \langle r_1, r_2 \rangle, \langle r_1, r_3 \rangle, \langle r_2, r_4 \rangle, \langle r_3, r_4 \rangle \}$ |

Relację RH możemy przedstawić graficznie w następujący sposób:



rys. 4.3. Relacja RH (hierarchia ról).

Zgodnie z definicją funkcji $roles(s_i)$ z $RBAC_1$ otrzymujemy, że sesje użytkowników: u_1 , u_2 , u_3 , u_4 , u_5 , są powiązane odpowiednio z rolami:

u_1 : r_1 ,
 u_2 : r_1, r_2 ,
 u_3 : r_1, r_3 ,
 u_4 : r_1, r_2, r_3, r_4

oraz przysługują im następujące zbiory praw

p_1 : p_1 ,
 p_2 : p_1, p_2, p_3 ,
 p_3 : p_1, p_4 ,
 p_4 : p_1, p_2, p_3, p_4, p_5 .

c) $RBAC_2$

$RBAC_2$ jest rozszerzeniem $RBAC_0$ o pojęcie ograniczeń związanych z rolami. Ograniczenia, obok hierarchii są uznawane za najważniejszy element polityki RBAC. Są one odpowiedzialne na poziomie organizacji za ograniczenie możliwości wystąpienia nadużyć spowodowanych poprzez np. łączenie funkcji nadzorcy i nadzorowanego. Ograniczenia są bardzo silnym mechanizmem wspierającym wdrażanie polityki organizacji wysokiego poziomu [San96].

$RBAC_2$ jest tożsame z $RBAC_0$ za wyjątkiem wprowadzenia w tym modelu dodatkowego wymogu istnienia kolekcji ograniczeń, które decydują czy wartości poszczególnych komponentów w $RBAC_0$ są, czy nie są akceptowalne. Jedynie akceptowalne wartości będą dozwolone.

Ograniczenia przeważnie odnoszą się do relacji przypisania podmiotu do roli oraz do relacji przypisania prawa dostępu do roli.

Wyróżnione zostały trzy kategorie ograniczeń:

1. Wzajemnie wykluczające się role (ang. mutually exclusive roles).

Ograniczenie to polega na identyfikacji i odrzucaniu wzajemnie wykluczających się ról/uprawnień ze zbioru aktywnych dla danego podmiotu powiązań rola-podmiot. Oznacza to, że podmiot może być skojarzony tylko z jedną z ról należących do zbioru ról wzajemnie wykluczających się. To ograniczenie wspiera paradygmat bezpieczeństwa mówiący o rozdzielności obowiązków w systemie.

Przykładem mogą być role *nadzorca billingu* i *użytkownik systemu*. Przy założeniu, że użytkownik jest zainteresowany tym, aby koszty jego aktywności były jak najmniejsze, a nadzorca billingu ma moc zmiany zapisu wartości oznaczającego rzeczywisty koszt aktywności użytkownika w systemie, role te nie powinny być łączone przez jeden podmiot.

2. Ograniczenie na ilość aktywnych połączeń z rolą (ang. cardinality constraints).

To ograniczenie jest odpowiedzialne za sterowanie liczbą aktywnych podmiotów w przypadku określonych ról, niewrażliwych dla działania systemu i organizacji. Może ono zapobiegać przypadkowemu lub celowemu rozpowszechnianiu się uprawnień o dużej mocy.

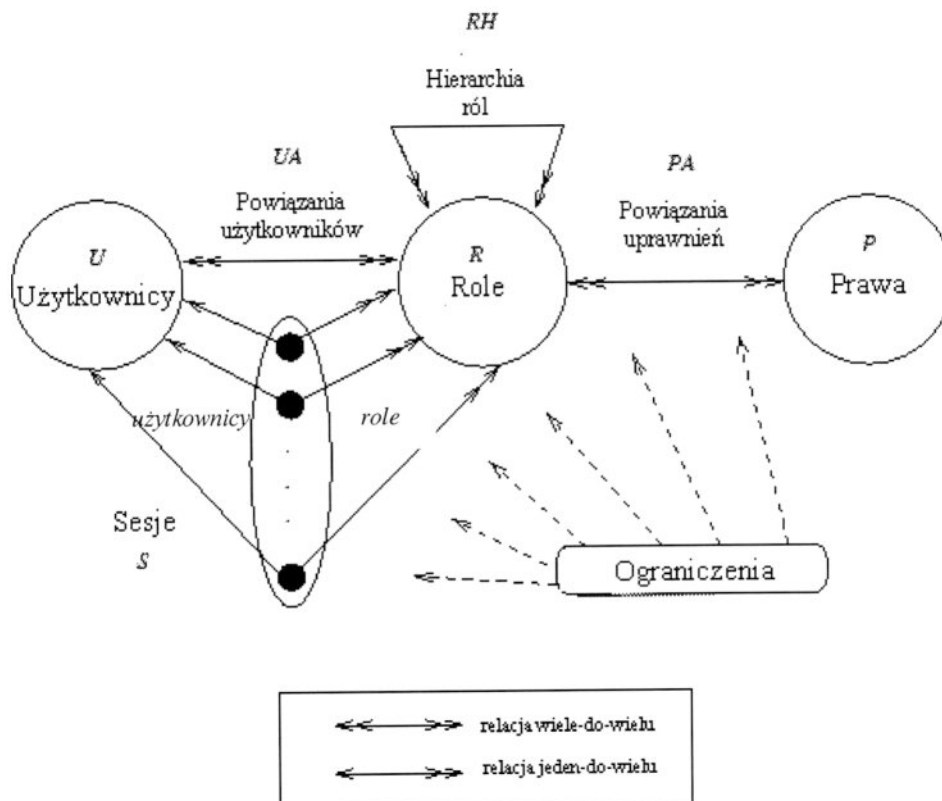
3. Role wzajemnie się wymagające (ang. prerequisite constraints).

Ograniczenia tego typu umożliwiają sterowanie warunkami wstępnymi niezbędnymi do aktywacji podmiotu w danej roli. Dzięki ograniczeniom tego typu można sterować np. przyznawaniem ról podmiotom gdy są one (nie są) już wcześniej członkiem określonej roli. Analogiczny warunek możemy odnieść do zarządzania uprawnieniami w obrębie danej roli.

Ograniczenia mogą odnosić się również do sesji, funkcji user oraz roles. Hierarchia ról może być również traktowana jako specyficzny rodzaj ograniczeń, istniejący pomiędzy rolami.

d) $RBAC_3$

$RBAC_3$ łączy w sobie cechy $RBAC_2$ i $RBAC_1$ czyli możliwość budowanie hierarchii ról, jak i opisywania dodatkowych ograniczeń określających relacje pomiędzy rolami. Istotną cechą tego modelu jest to, że pozwala on identyfikować istniejące sprzeczności pomiędzy hierarchią, a pozostałymi ograniczeniami zdefiniowanymi w systemie.



rys.4.3. Model $RBAC_3$

4.2.1.2. ARBAC'97

ARBAC'97 jest próbą wykorzystania polityk RBAC również i do zarządzania mechanizmem sterowania dostępem opartym o RBAC (w tym przypadku konkretnie chodzi o RBAC'96). Oznacza to, że mechanizmy RBAC zostają wykorzystane nie tylko do regulowania interakcji pomiędzy podmiotami i obiektami, ale również i do dynamicznego kształtowania polityki sterowania dostępem, poprzez ingerencję w zbiory R, P, PA, UA .

W tym celu definiuje się dodatkowe zbiory:

- AP - zbiór administracyjnych uprawnień, który umożliwia dokonanie zmiany elementów RBAC'96
- AR - zbiór ról administracyjnych

Dodatkowo zbiór AP powinien posiadać własność $AP \cap P = \emptyset$, zaś zbiór AR własność $AR \cap R = \emptyset$.

Role administracyjne mogą tworzyć hierarchię podobnie jak role regularne, hierarchia ta później jest mapowana na hierarchię ról regularnych., w celu wyznaczenia zakresów kompetencji pomiędzy poszczególnymi podmiotami administracyjnymi.

ARBAC'97 składa się z trzech części: URA97, PRA'97 i RRA'97.

a) URA'97, przypisanie typu podmiot-rola.

URA'97 odnosi się do zadania administracji relacją UA (podmiot-rola), zaś przypisywanie podmiotów do ról administracyjnych nie wchodzi w zakres URA'97. Z zadaniami URA'97 wiąże się potrzeba określenia dwóch następujących warunków:

- jakie role mogą być modyfikowane przez daną rolę administracyjną,
- jaki jest zbiór podmiotów, które mogą być członkami tych ról.

Warunki te wyrażone są w postaci wyrażeń z wykorzystaniem operatorów logicznych ' $\wedge$ ', ' $\vee$ ' oraz negacji ' $\neg$ '. Dla każdej roli administracyjnej określony jest zbiór CR warunków wstępnych odnośnie możliwości tworzenia przypisania podmiotu do roli z R.

Na potrzeby zapisu warunków wstępnych przyjęta jest następująca notacja:

$$\begin{aligned} x \text{ jeżeli } (\exists x' \geq x)(u, x') \in UA \\ \neg x \text{ jeżeli } (\forall x' \geq x)(u, x') \notin UA \end{aligned}$$

Jeżeli mamy określony następujące warunki wstępne i zakresy:

Rola administracyjna	Kierownik_Administracyjny
Warunek wstępny (zbiór CR)	$\text{Pracownik_działu_T} \wedge \neg \text{Kierownik_działu_T}$
Zakres	$[\text{Pracownik_działu_K}, \text{Kierownik_działu_k}]$

Zapis ten oznacza, że każdy podmiot będący członkiem roli administracyjnej *Kierownik_Administracyjny*, ma prawo każdemu podmiotowi wcześniej skojarzonemu z rolą *Pracownik_działu_T* i jednocześnie nie będącemu członkiem roli *Kierownik_działu_T* przyznać prawo do odgrywania również dowolnej roli z zakresu $[\text{Pracownik_działu_K}, \text{Kierownik_działu_k}]$

URA'97 określa możliwość przyznania/odwołania powiązania podmiot-rola na podstawie następującej relacji:

$$\text{can-assign} \subseteq \text{AR} \times \text{CR} \times 2^R \text{ i relacji } \text{can-revoke} \subseteq \text{AR} \times 2^R$$

Znaczenie $\text{can-assign}(x, y, z)$ jest następujące:

Podmiot będący członkiem roli administracyjnej x (lub roli starszej w hierarchii) może przypisać podmiot, którego obecny stan spełnia warunki wstępne określone w y , do roli w zakresie z .

Znaczenie $\text{can-revoke}(x, y)$:

Podmiot będący członkiem roli administracyjnej x (lub roli starsze w hierarchii) może odwołać podmiot z jakiegokolwiek roli w zakresie y .

Zaproponowany mechanizm odwoływania podmiotów z ról jest tzw. słabym odwołaniem. Zezwala on na zachowanie przynależności podmiotu do ewentualnie posiadanych przez niego ról nadrzędnych względem aktualnie odwoływanej roli.

b) PRA'97, przypisanie prawo-rola.

PRA'97 odnosi się do problemu przypisywania i odwoływania uprawnień do ról. Filozofia działania tego mechanizmu jest analogiczna do tej, która została zastosowana w URA'97, a różni się tym, że nie odnosi się do powiązania ról i podmiotów, a do relacji praw i ról.

Odwołanie/przypisanie do relacji prawo-rola jest autoryzowane na podstawie relacji:

$$\text{can-assignp} \subseteq \text{AR} \times \text{CR} \times 2^R \text{ i relacji } \text{can-revokep} \subseteq \text{AR} \times 2^R$$

Znaczenie tych relacji jest analogiczne do tych znanych z URA'97.

c) RRA'97, przypisanie rola-rola.

Sposób administrowania powiązaniem pomiędzy rolami przedstawiony przez R.Sandhu [SB97] nie może stanowić podstawy dla implementacji w rzeczywistych systemach, gdyż w znacznej mierze jest on nieformalny i oparty jedynie na intuicji poprawnego zarządzania.

Wśród wszystkich istniejących w systemie ról można wyróżnić trzy następujące kategorie:

1. Możliwości - role, które zawierają jedynie uprawnienia,
2. Grupy - role, które zawierają jedynie podmioty,
3. UP-role - role, które nie mają ograniczeń co do rodzaju przynależących do nich elementów, mogą zawierać podmioty, prawa, możliwości, grupy i inne UP-role (UPR - zbiór tych ról).

Problem zarządzania możliwościami jest analogiczny do problemu PRA'97 i przyjmuje się dla niego nazwę ARA'97 oraz następującą definicję:

Odwołanie/przypisanie do relacji rola-rola jest autoryzowane na podstawie relacji:

$$\text{can-assigna} \subseteq \text{AR} \times \text{CR} \times 2^{\text{UPR}} \text{ i relacji } \text{can-revokea} \subseteq \text{AR} \times 2^{\text{UPR}}$$

Problem zarządzania grupami jest analogiczny do problemu z URA'97 i przyjmuje się dla niego następującą definicję:

Odwołanie/przypisanie do relacji rola-rola jest autoryzowane na podstawie relacji:

$$\text{can-assigng} \subseteq \text{AR} \times \text{CR} \times 2^{\text{UPR}} \text{ i relacji } \text{can-revokeg} \subseteq \text{AR} \times 2^{\text{UPR}}$$

Dla problemu UP-ról przyjmuje się następującą definicję:

Tworzenie ról, usuwanie ról, dodawanie krawędzi rola-rola, usuwanie krawędzi rola-rola jest możliwe dla:

$$\text{can-modify: } \text{AR} \rightarrow 2^{\text{UPR}}$$

Znaczenie $\text{can-modify}(x,y)$ jest następujące:

- rola administracyjna (lub rola względem niej nadrzędna) jest uprawniona do tworzenia i usuwania ról w zakresie y z wyjątkiem punktów końcowych y i może modyfikować relacje pomiędzy rolami w zakresie y . Możliwości te są jednak ograniczone warunkiem, że wprowadzane zmiany relacji pomiędzy elementami hierarchii ról muszą zachowywać spójność całego układu.

4.2.1.3. Problemy RBAC'96 i ARBAC'97.

Przedstawiony model RBAC, zaproponowany przez R.Sandhu jest próbą kompleksowego podejścia do problemów związanych ze sterowaniem dostępem poprzez role. Wyróżnione dwa zasadnicze elementy tego modelu RBAC'96 i ARBAC'97 dotyczą dwóch podstawowych domen każdego systemu sterowania dostępem: sterowaniem interakcjami pomiędzy elementami pasywnymi i aktywnymi oraz sposobu zarządzania elementami systemu sterowania.

Przyjęta w RBAC'96 definicja roli stanowi uproszczenie ogólnej definicji, która np. dopuszcza istnienie obowiązków w zbiorze uprawnień skojarzonych z rolą [Jon93]. Kolejnym uproszczeniem jest rezygnacja z możliwości stosowania w obrębie roli zakazów (ang. negative permissions).

Oba te uproszczenia rzutują na możliwości jakie daje ten model na etapie przechodzenia z definicji ról na poziomie organizacji na role zdefiniowane w systemie sterowania dostępem. W najlepszym przypadku zakłócona zostanie intuicyjność przejścia pomiędzy tymi dwoma poziomami. W gorszym, niemożliwe będzie uchwycenie w systemie sterowania dostępem pełnej specyfiki zadania. Np. naturalnym elementem definicji zadań na poziomie organizacji jest pojęcie konieczności, obowiązku. Zazwyczaj pojęć tych nie da się dokładnie odtworzyć operując prostymi typami praw takich jak read, write, etc., które mówią jedynie o możliwości wykonania danych operacji.

Przyjęte założenia odnośnie zbioru praw określających zakres możliwości podmiotów w systemie uniemożliwiają sterowanie bazujące na sekwencji wykonywanych akcji, czy też uwzględniającą zależności czasowe pomiędzy wykonywanymi akcjami. Stanowić to może, poważne ograniczenie dla zastosowania tego mechanizmu np. w aplikacjach typu „work-flow”, w których istotnym elementem sterowania są wyżej wymienione zależności.

Mniej istotne dla możliwości implementacyjnych RBAC jest brak istnienia możliwości stosowania obok zezwoleń na liście praw skojarzonych z rolą, również i zakazów oraz obowiązków. Możliwość stosowania zakazów może usprawnić sposób sterowania rozdzielnością zadań, np. wprowadzony w sposób jawny do roli zakaz wykonywania określonej czynności, daje pewność, że podmiot nie uzyska tego prawa bez jednoczesnego pogwałcenia spójności uprawnień.

Elementem istniejącym w modelu RBAC'96, a który eliminuje część niedogodności związanych z brakiem zakazów, są ograniczenia z RBAC₁, które to mogą pełnić podobną funkcję przy definiowaniu polityki sterowania. Zaproponowany przez R.Sandhu sposób implementacji ograniczeń nie jest jednak pozbawiony wad. Najbardziej istotne jest to, że ograniczenia są wprowadzone jedynie w sposób nieformalny, który w praktyce wyklucza możliwość ich zastosowania. Poza tym, zdefiniowany zbiór ograniczeń odnosi się jedynie do relacji pomiędzy podmiotami występującymi w poszczególnych rolach, ominięto zaś problem sterowania zależnościami pomiędzy uprawnieniami przypisywanym rolom.

Abstrahując od technicznych możliwości ograniczeń zaproponowanego modelu, należy mieć na uwadze, że jakkolwiek mechanizm ograniczeń jest skuteczny jedynie w przypadku konsekwentnie i poprawnie funkcjonującej polityki przyznawania ról. Jeżeli podmiot będzie mógł występować w systemie w rozmaitych rolach to istnieje zawsze prawdopodobieństwo, że uda mu się ominąć ograniczenie związane z jedną rolą przy pomocy innej roli.

Również system zarządzania ARBAC'97 jest obciążony pewnymi wadami. Pierwszym zastrzeżeniem odnośnie tego modelu jest to, że nie została uwzględniona w nim możliwość tworzenia i usuwania podmiotów i praw. Jest to poważne ograniczenie, zważywszy na dynamikę większości organizacji oraz ich systemów informatycznych. W takim przypadku, konieczne byłoby zdefiniowanie podmiotów i praw w sposób nadmiarowy na początku działania systemu, co umożliwiłoby ich późniejsze wykorzystanie w miarę rosnących zapotrzebowań. Jednak rozwiązanie takie nie może być do zaakceptowania w przypadku większości istniejących systemów.

Kolejną niedogodnością ARBAC'97 jest to, że większość operacji zarządzania jest przedstawiana jedynie w sposób nieformalny. Wyklucza to w praktyce jakąkolwiek możliwość formalnego badania konsekwencji podejmowanych decyzji na własności polityki sterowania dostępem i stanu bezpieczeństwa systemu. Ta sama uwaga odnosi się również do definiowania procesu zarządzania rolami i ograniczeniami.

W świetle wyżej wymienionych problemów modelu mechanizmu RBAC proponowanego przez R. Sandhu, mniejszym problemem jest brak możliwości obsługi delegowania ról, tzn. tymczasowego zlecenia wykonania określonego zadania podmiotom, które nie jest bezpośrednio odpowiedzialny za jego realizację.

4.2.2. Model mechanizmu dla ograniczeń w polityce RBAC.

Celem zastosowania ograniczeń w RBAC była chęć odzwierciedlenia istniejących w politykach wyższego szczebla mechanizmów pozwalających odzwierciedlać dodatkowe, poza hierarchicznymi, zależności pomiędzy rolami. Zagadnienia związane z ograniczeniami i polityką RBAC stanowią główną część pracy Chen i Sandhu „Constraints for Role-Based Access Control” [ChS96].

W celu opisu ograniczeń i ich realizacji konieczne jest zdefiniowanie języka opisu i istnienie odpowiednich funkcji systemowych. Chen i Sandhu przyjęli, że język ten powinien być prosty i intuicyjny, tak aby mógł być stosowany bez potrzeby dodatkowych szkoleń. Powinien być również elastyczny i posiadać wystarczającą siłę ekspresji, aby móc sprostać aktualnym potrzebom systemu bezpieczeństwa. Równocześnie warunkiem koniecznym jest to, aby posiadał on wystarczająco silne podstawy formalne, które zagwarantują jego precyzję [San96].

Propozycja przedstawiona przez autorów oparta jest na teorii zbiorów i składa się z następujących elementów:

- zbiorów opisujących - role, prawa, użytkowników (podmioty),
- operatory rachunku zbiorów - $\cup$ (suma), $\cap$ (przecięcie), - (różnica),
- operatory relacji - $\supset, \supseteq, \subset, \subseteq, \in, \notin, \neq$
- operatory logiczne - $\neg, \wedge, \vee, \rightarrow$
- dwie funkcje zastępujące kwantyfikatory:
 - $\text{one_element: set} \rightarrow \text{element}$ (wybiera jeden element ze zbioru elementów),
 - $\text{allother: set} \rightarrow \text{set}$ (wybiera zbiór poprzez wybranie po kolei wszystkich elementów),
 oraz: $\{\text{one_element}(s)\} \cup \text{allother}(s) = s$.

W takim podejściu warunek pozwalający na sprawdzenie konfliktów pomiędzy rolami autorzy zdefiniowali w następujący sposób:

$$\text{one_element}(R) \in \text{role-set}(\text{one_element}(U)) \rightarrow \text{allother}(R) \cap \text{roleset}(\text{one_element}(U)) = \emptyset$$

gdzie:

R- zbiór ról,

U - zbiór użytkowników,

role-set - funkcja globalna mapująca użytkowników do zbioru jego ról

Zapis ten jest tożsamy zapisowi wyrażonemu w logice predykatów pierwszego rzędu

$$\forall r \in R \quad \forall u \in U \quad (r \in \text{role-set}(u) \rightarrow (R - \{r\}) \cap \text{role-set}(u) \neq \emptyset)$$

Sterowanie ograniczeniami w tym modelu posiada dwie zasadnicze fazy. Pierwsza faza obejmuje proces definiowania ograniczeń, które są zapisywane w zdefiniowanym przez autorów języku formalnym. Ograniczenia te mogą być zdefiniowane np. poprzez określenie warunków wstępnych koniecznych do uzyskania danej roli, czy też poprzez określenie zbiorów ról wzajemnie się wykluczających. Należy przy tym nadmienić, że język formalny służy jedynie do zapisywania ograniczeń, natomiast nie pozwala on rozstrzygać o poprawności przyjętych ustaleń, tzn. o zgodności nowododawanych elementów z elementami już istniejącymi. Druga faza polega na sprawdzaniu zgodności wywołań funkcji przypisania (odwołania) podmiotu do roli z przyjętymi założeniami, zapisanymi w warunkach wstępnych.

Przykład 4.2.

Dany mamy zbiór ról wzajemnie się wykluczających (tzn. dany użytkownik może być powiązany tylko z jedną rolą z tego zbioru) oraz zbiór użytkowników:

$$R = \{r_1, r_2, r_3, r_4\},$$
$$U = \{u_1, u_2, u_3, u_4\}.$$

Badanie zgodności z ograniczeniem wzajemnego wykluczania się będzie polegało na testowaniu warunku:

$$\text{one_element}(R) \rightarrow \text{allother}(R) \cap \text{roleset}(\text{one_element}(U)) = \emptyset$$

Kolejne wywołania funkcji $\text{one_element}(R)$ będą przyjmować wartości: $\{r_1, r_2, r_3, r_4\}$, zaś funkcja $\text{allother}(R)$ będzie zwracać odpowiednio wartości: $\{R-r_1, R-r_2, R-r_3, R-r_4\}$.

W związku z tym, jeżeli do roli u_1 została przypisana rola r_1 , to otrzymujemy następujące wartości wywołań funkcji:

$$\text{one_element}(R) = r_1$$

$$\text{one_element}(U) = u_1$$

Przy takich wartościach wywołań tych funkcji spełnienie warunku:

$$\text{allother}(R) \cap \text{roleset}(\text{one_element}(U)) = \emptyset$$

gwarantuje zgodność przypisań ról do użytkownika u_1 , z ograniczeniem wzajemnego wykluczania.

Ponieważ podobny test będzie może być wykonany dla dowolnych elementów zbiorów R i U , to w efekcie otrzymamy pewność, że przypisanie ról dla wszystkich użytkowników jest zgodne z przyjętym ograniczeniem.

4.2.2.1. Problemy.

Zaproponowany przez Chena i Sandhu model jest prosty w swych założeniach, lecz nie rozwiązuje wszystkich problemów związanych z ograniczeniami w polityce RBAC.

Umożliwia on co prawda formułowanie w sposób formalny prostych ograniczeń takich jak wzajemne wykluczanie się ról, lecz brak w nim jest możliwości kompleksowej weryfikacji spójności przyjmowanych założeń. Wszelkie zmiany wprowadzane w systemie odnośnie ról, praw z nimi związanych, listy podmiotów mogą mieć pośredni wpływ na zgodność stanu systemu z warunkami określonymi w ograniczeniach. Zaproponowany model nie daje możliwości sterowania uwzględniającego taką weryfikację. Autorzy prezentowanego modelu nie przedstawili propozycji algorytmów badania zgodności stanu systemu z istniejącymi i nowotworzonymi ograniczeniami. Powoduje to, że nie możemy ocenić w sposób jednoznaczny możliwości automatyzacji procesów zarządzania takim systemem ograniczeń. Kolejną niedogodnością tego rozwiązania są problemy z funkcjonowaniem tak

zdefiniowanych ograniczeń na poziomie operacji (transakcji). Przykładem tego rodzaju problemów może być ograniczenie nakładające wymóg istnienia jednego i tylko jednego użytkownika z uprawnieniami administratora w systemie. Każda próba zmiany administratora w systemie korzystającym z tego typu ograniczeń skończy się niepowodzeniem, gdyż transakcja: *zmiana administratora X na administratora Y*, zostanie rozbita na ciąg akcji elementarnych:

a) *odwołaj przypisanie X do roli administratora,*

b) *powiąż Y z rolą administratora*

lub

a) *powiąż Y z rolą administratora,*

b) *odwołaj przypisanie X do roli administratora,*

co w każdym przypadku spowoduje wykrycie niedozwolonej z przyjętym ograniczeniem akcji już w kroku a).

Również niemożliwe jest w omawianym powyżej modelu uchwycenie wszystkich możliwych typów ograniczeń, a związanych np. z sekwencyjnością operacji, czasem wykonania, itp.

4.2.3. Model referencyjny mechanizmu RBAC.

W tym punkcie omówiony zostanie inny model mechanizmu dla polityki RBAC, który został zaproponowany przez D.F.Ferraiolo, J.F.Barkley i D.R. Kuhn [FBK95]. Model ten składa się z dwóch części nazwanych odpowiednio: MC_0 i MC_1 .

MC_0 jest modelem bazy danych autoryzacji, definiującym własności bezpieczeństwa procesu autoryzacji w postaci statycznych ról. MC_1 jest zaś modelem aktywacji, definiującym warunki dynamicznej aktywacji ról. MC_1 wymaga MC_0 , a oba modele składają się z następujących elementów:

- | | | |
|------------------|---|---|
| zbioru typów | - | definiuje on podstawowe typy elementów wraz z określonymi na nich funkcjami. Funkcje są dwójakiego rodzaju: mapujące (pozwalają określić zależności pomiędzy elementami) oraz ograniczeń (odzwierciedlają ograniczenia w relacjach pomiędzy elementami) |
| zbioru praw | - | reprezentuje założenia modelu |
| zbioru własności | - | określa sens praw i konsekwencje z nich wypływające |

MC_0 jest określony przez funkcje mapujące i statyczne właściwości, które opisują takie elementy modelu jak: hierarchia ról, dziedziczenie, statyczna rozdzielność zadań, itp.

Powiązanie roli z użytkownikiem (podmiotem) jest zdefiniowane poprzez funkcje:

- | | | |
|--|---|--|
| type | - | user, role (identyfikują użytkownika i nazwy ról), |
| RM (r: role) $\rightarrow 2^{\text{user}}$ | - | funkcja mapująca, autoryzuje użytkownika w roli r |

Powiązanie zezwolenia z rolą:

type	- object (obiekty, zasoby podlegające ochronie, takie jak pliki, kolejki, miejsce na dysku, itp.)
type	- operation (wyróżnione w systemie operacje odzwierciedlające charakterystyczny dla systemu zbiór aktywności, który może być wykonywany przez podmioty na obiektach)
type	- permission = $2^{\text{operation} \times \text{object}}$
$RP(r:\text{role}) \rightarrow 2^{\text{permission}}$	- funkcja mapująca, autoryzuje użytkownika w roli r
$POp(p:\text{permission}) \rightarrow \{\text{operation}\}$	- powiązanie praw z operacją
$POb(p:\text{permission}) \rightarrow \{\text{object}\}$	- powiązanie praw z obiektem

Wprowadza się pojęcie zawierania ról, którego idea jest zbliżona do koncepcji dziedziczenia ze środowisk zorientowanych obiektowo. Definiuje się je formalnie w następujący sposób:

$$(\forall i, j: \text{role})(\forall u: \text{user}): \supseteq j \wedge u \in RM[i] \Rightarrow u \in RM[j]$$

Kompletny zbiór relacji zawierania dla danego zbioru ról jest nazwany hierarchią ról.

Statyczna separacja obowiązków jest zdefiniowana w następujący sposób:

$Ea: \text{role} \times \text{role}$ - zbiór wykluczających się par ról

$$(\forall i, j: \text{role})(\forall u: \text{user}) u \in RM[i] \wedge u \in RM[j] \Rightarrow (i, j) \in Ea$$

Ograniczenia podlegają również dziedziczeniu w hierarchii.

Ograniczenie ilości wystąpień danej roli:

$$(\forall r: \text{role}) RL[r] \geq RM[r]$$

gdzie:

$ML(r: \text{role})$ - liczba podmiotów mogących posiadać autoryzację do roli r.

Dodatkowo dla modelu MC_0 zdefiniowano i udowodniono następujące twierdzenia [FBK95]:

1. *Twierdzenie o spójności statycznej i separacji obowiązków i zawierania ról.*

Dwie niepuste role i oraz j $RM[i] \neq 0$, $RM[j] \neq 0$ mogą być w relacji wzajemnego wykluczania się $\Leftrightarrow$ są związane relacją dziedziczenia w hierarchii ról.

$$(i, j) \in Ea \Rightarrow \neg(i \supseteq j \vee j \supseteq i)$$

2. *Wnioskowanie o spójności zawierających się ról.*

Niepusta rola nie może być w relacji wzajemnego wykluczania się z samą sobą.

$$\forall i: (i,i) \in E_a$$

3. *Twierdzenie o niedziedziczności wzajemnie wykluczających się ról*

Jeżeli istnieje para $(i,j) \in E_a$, to nie może istnieć taka niepusta rola k , że $K \supseteq i \wedge k \supseteq j$

4. *Wniosek o nieistnieniu roli superużytkownika.*

Jeżeli istnieje para $(i,j) \in E_a$, wtedy nie może istnieć niepusta rola r taka, że dla wszystkich i $r \supseteq i$.

5. *Twierdzenie o przechodniości wzajemnego wykluczania się.*

Jeżeli użytkownik dziedziczy rolę, która jest w relacji wzajemnego wykluczania się z jakąkolwiek inną rolą, to użytkownik nie może być autoryzowany dla tej drugiej roli.

$$u \in RM[i] \wedge i \supseteq j \wedge (j,k) \in E_a \Rightarrow u \notin RM[k]$$

6. *Twierdzenie o dziedziczeniu ograniczeń statycznych.*

Dla ról i,j jeżeli $i \supseteq j$ oraz $C(RM[j])$, gdzie C jest pewnym ograniczeniem na użytkownika autoryzowanego dla roli j , wtedy $C(RM[i])$

Badanie warunków spójności odbywa się na zasadzie sprawdzania zgodności z powyższymi twierdzeniem bieżącego stanu systemu przez odpowiednio zaimplementowane procedury testujące.

Model MC_1 dotyczy opisu dynamicznych własności RBAC, uzupełnia MC_0 o aktywację ról, egzekucję praw, dynamiczną rozdzielność obowiązków i dostępu do obiektów.

Dynamiczne właściwości RBAC pozwalają na bardziej precyzyjną realizację zasady minimalnych uprawnień np. poprzez czasowe blokowanie, udzielanie dodatkowych uprawnień.

Podstawowym pojęciem w tym modelu jest podmiot, który jest aktywną jednostką wykonującą akcje w systemie zlecone przez użytkownika

type subject	-	zbiór identyfikujący podmioty
SU(s:subject)→user	-	funkcja mapująca podmiotów na użytkowników, zwraca użytkownika powiązanego z podmiotem s
AR(s:subject)→2 ^{role}	-	funkcja zwraca zbiór ról w których podmiot jest aktywny

$$(\forall s:\text{subject}, u:\text{user}, r:\text{roles}) r \in AR[s] \Rightarrow SU[s] \in RM[r]$$

Podmiot nie może być aktywny w roli, która nie jest autoryzowana dla powiązanego z nią użytkownika.

Dynamiczna rozdzielnosc obowiazkow jest zdefiniowana formalnie w nastepujacy sposob:

$$(\forall s,t:\text{subject})(\forall i,j:\text{role}): s \neq t \text{ oraz} \\ i \in \text{AR}[s] \wedge j \in \text{AR}[t] \wedge (i,j) \in \text{Er} \Rightarrow \text{SU}[s] \neq \text{SU}[t]$$

gdzie:

$\text{Er}:\text{role} \times \text{role}$ - zbiór par ról będących ze sobą w konflikcie w trakcie ich aktywności, tzn. żaden z użytkowników nie może korzystać jednocześnie z uprawnień obu tych ról.

W przeciwieństwie do ograniczeń statycznych, dynamiczne mogą być hierarchicznie powiązane przez relację zawierania. Jest to możliwe dzięki rozróżnieniu operacji autoryzacji i aktywacji.

Podmiot może wykonać daną operację tylko wtedy gdy ta operacja jest autoryzowana dla roli w której w danej chwili podmiot jest aktywny.

$$(\forall s:\text{subject})(\forall op:\text{operation}) \\ \text{exec}(s,op) \Rightarrow (\exists r:\text{role})(\exists p:\text{permission}) r \in \text{AR}[s] \wedge p \in \text{RP}[r] \wedge op \in \text{POp}[p]$$

gdzie: $\text{exec}:\text{subject} \times \text{operation} \rightarrow \text{boolean}$

$$\text{exec}(s,op) = \begin{cases} 1 & \text{jeżeli } s \text{ może wykonywać } op \\ 0 & \text{w przeciwnym wypadku} \end{cases}$$

Podmiot może wykonać operację na obiekcie tylko wtedy gdy istnieje taka rola, która jest elementem zbioru aktywności tego podmiotu i rola ta zawiera prawo, które autoryzuje wykonanie tej operacji na obiekcie.

$$(\forall s:\text{subject})(\forall o:\text{object})(\forall op:\text{operation}) \\ \text{access}(s,p,op) \Rightarrow (\exists r:\text{role})(\exists p:\text{permission}) r \in \text{AR}[s] \wedge p \in \text{RP}[r] \wedge op \in \text{POp}[p] \wedge o \in \text{POb}[p]$$

gdzie: $\text{access}:\text{subject} \times \text{object} \times \text{operation} \rightarrow \text{boolean}$

$$\text{access}(s,op,o) = \begin{cases} 1 & \text{jeżeli } s \text{ ma dostęp do obiektu } o \text{ poprzez operację } op \\ 0 & \text{w przeciwnym wypadku} \end{cases}$$

4.2.3.1. Problemy

Zasadniczym problemem związanym z powyżej opisanym modelem mechanizmu polityki RBAC jest brak rozwiązań z zakresu zarządzania sterowaniem dostępem. Model ten pozwala jedynie na opisanie zasad sterowania dostępem podmiotów do obiektów, z wykluczeniem obiektów systemowych zawierających informację na temat ról, hierarchii, ograniczeń, itp. Domyślnym założeniem przyjętym przez autorów tego modelu jest to, iż RBAC jest konfigurowany w początkowym stadium funkcjonowania systemu i jego konfiguracja nie ulega już żadnym dalszym zmianom w późniejszym czasie. Za jej prawidłowe określenie i odzwierciedlenie polityki organizacji w strukturach sterowania dostępem odpowiedzialna jest jedna osoba, a więc mamy w tym przypadku przykład scentralizowanego podejścia do zarządzania RBAC. W związku z tym, nie jest również możliwa decentralizacja funkcji administratora, nie jest to jednak zasadnicza wada przy wcześniej wspomnianym założeniu, o niezmienności struktury RBAC.

Nie jest również możliwe określenie ograniczeń związanych z definiowanymi rolami. Opisane ograniczenia zostały zdefiniowane jedynie na potrzeby modelowania wzajemnie wykluczających się ról i ograniczenia maksymalnej liczby podmiotów skojarzonych z daną rolą. Kontrola spójności, zaproponowana przez autorów, ograniczona jest do przeglądania statycznych tablic określających strukturę sterowania dostępem (np. tablice Er i Ea). Warunkiem koniecznym dla właściwego funkcjonowania systemu, jest zatem poprawne zdefiniowanie tych warunków. Niemożliwe jest badanie spójności w oparciu o analizę zgodności dodawanego uprawnienia ze zbiorem uprawnień do którego jest ona dołączane. Brak jest również ścisłej definicji pojęcia operacji, a przez to nie można stwierdzić, czy autorzy przedstawione rozważania prowadzili jedynie dla elementarnych akcji dostępu (read, write), czy też dopuszczali możliwość sterowania przez ich mechanizm złożonymi operacjami typu transakcji. W przypadku transakcji należałoby jednak przedstawić założenia, które określają stopień szczegółowości sterowania oraz warunki niezawodności realizacji samej transakcji.

4.2.4. Model obiektowo zorientowanego RBAC.

J. Barkley zaproponował dla polityki RBAC mechanizm oparty na technologii obiektowej. Wyróżnioną przez Barkley zaletą RBAC, wokół której skoncentrował swoje rozważania, jest możliwość definiowania operacji będących przedmiotem sterowania, w sposób jak najbardziej ogólny, pozwalający osiągnąć dużą elastyczność na poziomie administracji systemem jak i aplikacji [Bar95b].

Aplikacje działające w systemie odwołują się do jego zasobów poprzez aktywację wybranych operacji ze zdefiniowanego dla tego systemu zbioru operacji elementarnych. Jeżeli zbiór ten ulegnie zmianie, bądź ulegnie zmianie sposób odwoływania się do istniejących operacji, wymagane będzie dostosowania do nowych warunków całej aplikacji. Mechanizm zaproponowany przez Barkley ma na celu wyeliminowanie konieczności dostosowywania aplikacji do ograniczeń dostępu związanych z rolami wyróżnionymi w systemie i organizacji.

Wprowadza on pomiędzy poziom aplikacji i poziom informacji trzy dodatkowe poziomy klas:

- Klasy podstawowych metod dostępu (Basic Access Methods Class),
- Klasy roli (Role Class),
- Klasy interfejsu aplikacji (Application Interface Class).

Klasy podstawowych metod dostępu zawierają metody, które określają zbiór elementarnych operacji możliwych do wykonania w systemie przez podmioty względem obiektów systemowych. Sterowanie dostępem do tych metod, a przez to i do obiektów, jest organizowana na poziomie klas ról. Jedna klasa jest definiowana dla każdej wyróżnionej roli. Metody klas ról posiadają nazwy identyczne z metodami zdefiniowanymi w klasach podstawowych metod dostępu, z tym że ich zachowanie uwarunkowane jest poprzez ograniczenia narzucone na tę rolę przez politykę bezpieczeństwa. W zależności czy dana rola posiada możliwość korzystania z danej operacji czy nie, metoda roli wywołuje metodę z klasy podstawowych metod dostępu, bądź dokonuje odpowiednich zabiegów filtrowania odwołań przepływu informacji.

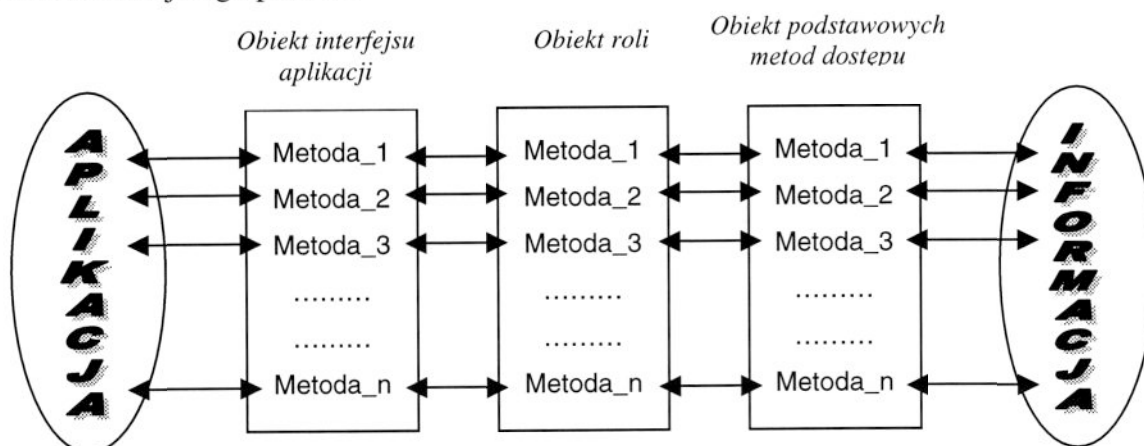
Metody klasy interfejsu aplikacji posiadają taką samą deklarację jak metody klasy ról. Metody obiektów tej klasy są aktywowane przez aplikacje. W zależności od aktualnej roli użytkownika, obiekt interfejsu aplikacji jest skojarzony z odpowiednim obiektem roli.

4.2.4.1. Problemy

Zaletą takiego podejścia do RBAC jest możliwość stosunkowo prostego osiągnięcia rozdzielności ról na poziomie aplikacji, bez konieczności bezpośredniej ingerencji w kod aplikacji. Trójstopniowy model zapewnia możliwość adaptacji sterowania przy jednoczesnym stałym interfejsie odwołań dla aplikacji oraz braku ingerencji w zbiór elementarnych operacji, które są bezpośrednio odpowiedzialne za dostęp do zasobów.

Wadą zaś propozycji Barkleya jest powierzchowność podejścia do polityki RBAC. Brak jest jakiegokolwiek dyskusji nad możliwością budowania hierarchii ról oraz związanymi z tym zagadnieniem problemami takimi jak dziedziczenie uprawnień, ograniczenia statyczne i dynamiczne pomiędzy rolami, kontrola spójności itp.

Nie przedstawiono również sposobu administrowania rolami oraz ewentualnej możliwości decentralizacji tego procesu.



rys.4.4. Mechanizm obiektowy polityki RBAC

4.3. Podsumowanie.

Przedstawione powyżej modele polityki RBAC ukazują możliwość różnorodnego podejścia do problemu implementacji tej polityki. Stanowią one również zobrazowanie podstawowych problemów, na które napotyka się w trakcie próby implementacji polityki sterowania dostępem poprzez role w systemie informatycznym. Na podstawie przeprowadzonej analizy można stwierdzić, że żaden z omówionych mechanizmów nie wypełnia wszystkich podstawowych wymagań stawianych przed systemami sterowania dostępem opartymi o politykę RBAC, a w tym wymagań określonych w pierwszym rozdziale tej pracy. Głównymi wadami omawianych modeli w kontekście wymagań sformułowanych w pierwszym rozdziale jest:

- brak istnienia odpowiednich mechanizmów pozwalających na jednoznaczne orzeczenie o zgodności lub niezgodności danego żądania dostępu z przyjętą polityką,
- problem z zarządzaniem systemem sterowania dostępem,
- brak możliwości automatyzacji sterowania dostępem.

Również i inne modele związane z tą polityką, które są opisywane we współczesnej literaturze nie zapewniają kompleksowego rozwiązania zidentyfikowanych w tym rozdziale problemów [Bar97] [BCFGK97] [FB97] [FK92] [FK95] [Gui95] [Kuh97] [NO95] [SCFY94] [SM98]. Dlatego też, można postawić tezę, że polityka RBAC wciąż potrzebuje zdefiniowania nowego modelu, pozwalającego w sposób pełny lub przynajmniej bardziej pełny od dotychczasowych rozwiązań, realizować jej założenia scharakteryzowane na początku tego rozdziału i wypełniać podstawowe wymagania związane ze współczesnymi systemami sterowania dostępem, które zostały omówione w pierwszym rozdziale tej pracy. Opracowanie takiego modelu przyczyniłoby się do zwiększenia efektywności sterowania dostępem oraz zarządzania dostępem w oparciu o politykę RBAC.

W nawiązaniu do przeprowadzonej w tym rozdziale analizy oraz do przedstawionych założeń polityki RBAC, sformułowano poniższą listę szczegółowych wymagań, których wypełnienie ma wpływ na usprawnienie działania mechanizmów związanych z polityką RBAC, a przez to na wypełnienie ogólnych wymagań sformułowanych w rozdziale pierwszym.

1. Model mechanizmu.

Warunek ten można określić jako przedwstępny, gdyż jest to wymaganie związane nie tylko z polityką RBAC ale i jakimkolwiek innym elementem systemu bezpieczeństwa systemu informatycznego. Formalne podejście umożliwia badanie i dowodzenie własności implementowanego mechanizmu, a przez to i wnioskowanie o jego bezpieczeństwie. Stwarza ono również możliwość automatyzacji pewnych procesów z zakresu zarządzania sterowaniem dostępem. Wymóg ten jest zapisany w dokumentach *Common Criteria* i *The Orange Book*, które stanowią podstawę oceny poziomu bezpieczeństwa współczesnych systemów.

2. Realizacja sterowania dostępem w zgodzie z założeniami polityki RBAC.

- weryfikacja zgodności żądań dostępu podmiotu z określonymi w jego roli uprawnieniami oraz z uprawnieniami przysługującymi mu na zasadzie dziedziczenia i delegacji.

Mechanizm powinien zapewniać, że zgłoszone przez podmiot żądanie dostępu zostanie zaakceptowane jedynie wtedy, gdy jest zgodne z przyjętą polityką, która jest określona przez zdefiniowane w systemie role i ich hierarchię.

3. Zarządzanie dostępem poprzez mechanizmy:

a) definiowania ról (rola- {prawa, zakazy, obowiązki}),

b) przypisywania podmiotów do ról (podmiot-rola):

- stałe,
- delegowane,

c) definiowania powiązań pomiędzy rolami (rola-rola):

- hierarchii ról,
- ograniczeń:
- wzajemne wykluczanie się ról,
- wzajemne wymaganie się ról,
- ograniczenie na maksymalną liczbę aktywnych podmiotów przypisanych do roli.

W tym szczególnie istotne jest aby model dawał możliwość kontrolowania spójności uprawnień przypisanych do określonej roli, spójności hierarchii ról i uprawnień przysługujących podmiotowi na podstawie delegacji roli.

Powyższa lista wymagań stanowiła punkt wyjścia przy konstruowaniu nowego modelu mechanizmu dla polityki sterowania i zarządzania dostępem poprzez role opartego o formalizm logiki deontycznej, który to zostanie przedstawiony w następnych rozdziałach.

5. Model sterowania i zarządzania dostępem oparty o politykę RBAC.

W tym rozdziale zostanie przedstawiony model sterowania dostępem zgodny z podstawowymi założeniami polityki RBAC. Model ten wykorzystuje formalizm logiki matematycznej, dzięki któremu możliwe jest ściśle definiowanie warunków sterowania oraz analiza konsekwencji wpływających z przyjętych unormowań.

Na początku rozdziału przedstawiono krótkie uzasadnienie wyboru logiki deontycznej jako formalizmu opisu polityki sterowania dostępem. Pokróćce omówiono istotne dla prezentowanego modelu zagadnienia związane z tą logiką jak i logikami modalnymi w ogóle. Następnie zdefiniowano model, który służy do modelowania polityki sterowania dostępem dla dowolnego systemu informatycznego.

5.1. Logika formalna.

Logika formalna jest teorią, która bada strukturę zdań oraz wnioskowania dedukcyjnego, abstrahując od treści zdań, a zajmując się jedynie ich formą logiczną.

Najwęższe rozumienie logiki redukuje ją do logiki predykatów pierwszego rzędu, zawierającej logikę zdań. Ta ze względu na swoje właściwości często jest przedstawiana jako odrębna teoria. Takie zakreślenie granic logiki może być uzasadnione faktem, że logika predykatów pierwszego rzędu wystarczy jako podstawa formalizacji rozumowań matematycznych. Otwarta jednak pozostaje kwestia tego, czy byłaby ona wystarczająca dla formalizacji wszystkich rozumowań przeprowadzanych w języku naturalnym [Mar87][Bor77].

Klasyczna logika predykatów charakteryzuje się dwoma następującymi założeniami:

- a) jedynymi jej stałymi logicznymi są funktory prawdziwościowe,
- b) występują w niej jedynie dwie wartości logiczne: Prawda i Fałsz.

Funktory prawdziwościowe (ekstensjonalne) cechuje to, że wartość logiczna zdania złożonego zależy jedynie od wartości logicznych zdań składowych. To znaczy, że istnieje wielo-jednoznaczna odpowiedniość między wartością logiczną zdania złożonego, a jego składnikami [Mar87].

W logikach modalnych nie obowiązuje założenie a) z klasycznej logiki pierwszego rzędu, tzn. występują w nich funktory modalne (intensjonalne). Funktor F jest funktorem modalnym działającym na zdanie A wtedy i tylko wtedy, gdy wartość logiczna $F(A)$ nie jest określona (a przynajmniej nie jest określona w sposób całkowity) przez wartość logiczną zdania A [Mar87].

Początki logik modalnych sięgają sylogistyki modalnej Arystotelesa oraz megarejskich stoickich teorii modalności. Współczesna historia logiki modalnej zaczyna się od pracy Lewisa, nawiązującej do idei implikacji u McColla. Wśród logik modalnych znajdujących największe zainteresowanie we współczesnych badaniach można wymienić takie jak:

- logiki epistemiczne (związane z pojęciem wiedzy),
- logiki deontyczne (związane z pojęciem obowiązku),
- logiki temporalne (związane z pojęciem czasu).

Logiki modalne najczęściej tworzy się poprzez dodanie do języka logiki klasycznej odpowiednich funktorów modalnych.

W logikach intuicjonistycznych i wielowartościowych nie obowiązuje założenie *b)* z logiki klasycznej. Logika intuicjonistyczna ma odzwierciedlać intuicjonistyczne metody dowodu, kodyfikować intuicjonistycznie dopuszczalne metody argumentacji. Formalną różnicą tej kategorii logik jest to, że odrzucają one niektóre prawa logiki klasycznej jako nie uzasadnione, np. prawo wyłączonego środka. [Mar87]

Logiki wielowartościowe dopuszczają możliwość występowania więcej niż dwóch wartości logicznych. Przykładem tego typu logiki może być trójwartościowy rachunek zdań Łukasiewicza. [Mar87]

5.1.1. Normatywny aspekt sterowania dostępem.

Zgodnie z przyjętą definicją, sterowanie dostępem jest procesem ograniczania praw lub możliwości podmiotów do komunikowania się z innymi podmiotami lub do aktywacji funkcji i usług w systemie komputerowym lub sieci. Podczas opracowywania polityki sterowania dostępem, tworzony jest zbiór praw, które przysługiwać będą poszczególnym podmiotom przy odwoływaniu się do zasobów systemu. Prawa te w najprostszej formie mogą być kombinacją elementarnych praw dostępu, zawierających się w zbiorze:

{odczyt, zapis, tworzenie, kasowanie}.

Możliwe jest również definiowanie i operowanie przez politykę sterowania dostępem na prawach będących złożeniem ciągu elementarnych akcji dostępu, a będących opisem dostępnych dla podmiotów w danym systemie transakcji, zawierających się np. w poniższym zbiorze operacji:

{twórz_kopię_zapasową, synchronizuj_dane, twórz_nowego_użytkownika}.

Podstawowy sens zbioru uprawnień zdefiniowanego w ten czy też inny sposób pozostaje jednak niezmienny - podmiot jest dopuszczony do wykonywania dowolnej operacji ze skojarzonego z nim na etapie procesu autoryzacji zbioru.

W związku z powyższym, pierwsze mechanizmy sterowania dostępem tworzone były jako proste zbiory wyliczeniowe uprawnień, z którymi kojarzone były odpowiednie podmioty bądź grupy podmiotów. Była to metoda prosta, a przez to również i efektywna, dlatego większość systemów sterowania dostępem tworzonych w latach 80-tych i na początku lat 90-tych opierała się właśnie na mechanizmach tego typu [Lea90], [Lea94], [NCSC85], [ISO99].

Wraz z rozwojem systemów informatycznych oraz zwiększeniem ilości zasobów pozostających w obrębie kontroli danego systemu, takie proste rozwiązania przestały być wystarczające. Powszechna stała się również konieczność sterowania nie tylko elementarnymi akcjami dostępu typu zapis/odczyt, ale i transakcjami, będącymi sekwencjami elementarnych akcji dostępu, którym należy zapewnić odpowiednie warunki wykonania (np. niepodzielność).

Współczesne systemy wymagają aby nie tylko można było w dowolny sposób definiować zbiór praw dla poszczególnych podmiotów, ale i aby możliwe było weryfikowanie logicznej poprawności tworzonych zbiorów uprawnień. Weryfikacja ta ma na celu udzielenie odpowiedzi m.in. na pytanie czy zbiór ten jest zgodny z zasadą minimalnych uprawnień. W przypadku gdy przyznane podmiotowi prawa nie są jedynie elementarnymi akcjami dostępu, konieczne staje się również weryfikowanie spójności danego zbioru praw, tzn. zagwarantowanie, że dany podmiot nie będzie w posiadaniu dwóch sprzecznych ze sobą uprawnień [ABN89], [CC95], [CC97], [Cup95], [Kol98], [Kol99a].

Kolejną cechą charakterystyczną dla współczesnych systemów informatycznych jest dążenie do tego by wszystkie czynności związane z zarządzaniem systemem były możliwe do wykonania na jak najwyższym poziomie abstrakcji, tzn. aby były one jak najbardziej zbliżone do sposobu zarządzania na poziomie organizacji, bez konieczności zagłębiania się w szczegóły implementacyjne systemu operacyjnego, sprzętu, itp. W dziedzinie sterowania dostępem oznacza to konieczność stworzenia odpowiedniego algorytmu na podstawie opisu wyrażonego w języku naturalnym, który zawiera pojęcia m.in. tego co jest dozwolone, co jest zakazane, a co jest obowiązkowe dla danego podmiotu (grupy podmiotów) [CS96], [CD96], [GE90], [FT98], [BC91a], [RF98], [Por77].

Analizując język w którym wyrażana jest polityka sterowania dostępem, możemy zauważyć, iż jest on ściśle związany z takimi pojęciami jak: *jest dozwolone*, *jest zakazane*, *jest obowiązkowe*. Z drugiej strony, mamy formalizm logiki deontycznej (gr. *δέον* - powinność, słuszność), która to jest systemem dedukcyjnym z twierdzeniami zawierającymi stałe deontyczne o następującym znaczeniu: *jest obowiązkowe, że (O)*, *jest dozwolone, że (P)* oraz *jest zakazane, że (F)*. Istnieje wiele rozmaitych systemów logiki deontycznej, które różnią się przede wszystkim interpretacją znaczenia przypisywanego stałym deontycznym, a przez to różnią się również aksjomatyką. Szersze omówienie problematyki związanej z logiką deontyczną oraz jej historią można znaleźć w np. w pracach [BDRS95], [BDRS98], [Hil81], [Suc80], [Zie68], [Zie69].

Za zastosowaniem formalizmu logiki deontycznej w mechanizmach sterowania dostępem systemów informatycznych może przemawiać również fakt, że była ona wykorzystywana z powodzeniem przy analizie norm prawnych, polityk i opisów systemów [Kan72], [Suc80], [JS93], [WM93], [Zie69].

Analogicznie jak w tamtych zastosowaniach, system uregulowań związanych ze sterowaniem dostępem może być postrzegany jako zbiór norm wyznaczający zakres dozwolonych dla danego podmiotu akcji. Dlatego też, powinno być możliwe analizowanie spójności poszczególnych elementów polityki sterowania dostępem z całościową polityką, tak samo jak ma to miejsce w przypadku analizy spójności danego systemu prawnego. Zastosowania prawnicze logiki deontycznej zwracają uwagę na jeszcze jeden istotny aspekt formalnego opisu statusu danej aktywności, a mianowicie na możliwość badania konsekwencji z wykroczenia przeciw obowiązującym normom. W dziedzinie prawa w takim przypadku mówimy o stosowaniu sankcji przeciw podmiotowi przekraczającemu ustalone normy. W przypadku zastosowań informatycznych, otwiera to możliwość np. analizy stanów awarii systemu, tzn. takich w których z jakichkolwiek przyczyn nie są spełnione pierwotnie obowiązujące założenia [FT98], [PS94].

Najważniejszą jednak zaletą logiki deontycznej jest to, że jest to konkretny przykład logiki matematycznej, a przez to jest z nią również związany cały aparat formalizmów służących analizie wyrażeń formułowanych w jej języku. Daje to nam m.in. możliwość weryfikowania poprawności twierdzeń definiowanych w języku zbliżonym do języka naturalnego, a wyrażającego politykę sterowania dostępem. Aby móc wykorzystać możliwie wiele zalet formalizmu logiki, musimy zatem precyzyjnie zdefiniować język oraz mechanizmy wnioskowania pozwalających na automatyzację procesu analizy.

W dalszej części tego rozdziału przedstawiona zostanie propozycja języka i mechanizmów wnioskowania pozwalających na usprawnienie procesu sterowania i zarządzania polityką dostępu.

5.2. Definicja syntaktyczna i semantyczna języka modelu RMM.

Pierwszym krokiem na drodze do formalizacji sterowania i zarządzania dostępem jest zdefiniowanie języka opisu.

Proponowany przeze mnie model jest wariantem tzw. logiki *QPL* będącej rozszerzeniem logiki klasycznej pierwszego rzędu (*FOL*) o normalną zdaniową logikę modalną, której reguły i aksjomaty stosują się również do formuł z kwantyfikatorami [BDRS95]. Logika *QPL* min. zakłada zasadę *zawierania dziedzin*, oraz *zasadę luk w wartościach logicznych*.

Definicja 5.1. ZASADA ZAWIERANIA DZIEDZIN.

„jeśli świat u jest widoczny ze świata w , to dziedzina w jest podzbiorem dziedziny u ”

Definicja 5.2. ZASADA LUK W WARTOŚCIACH LOGICZNYCH.

„ $\varphi(t)$ ma wartość logiczną wtedy i tylko wtedy, gdy t ma określoną wartość”.

Poniżej przedstawiona jest definicja syntaktyczna modalnego języka opisu polityki sterowania dostępem poprzez role modelu *RMM* (*RBAC Modal Model*).

Definicja 5.3. SYNTAKTYKA JĘZYKA MODELU *RMM*.

Słownik języka *RMM* składa się z następujących rodzajów wyrażeń:

- a) zmiennych zdaniowych:
 $p \ q \ r \ p_1 \ q_1 \ \dots$
- b) zmiennych indywiduowych:
 $x \ y \ z \ x_1 \ y_1 \ z_1 \ \dots$
- c) n -argumentowych stałych predykatowych ($n=1,2,3,\dots$):
Poziom(x,y), *Utwórz*(x,y), *Usuń*(x), *Poziom_dziecka*(x,y), *Dodaj_dziedzicz*(x,y),
Grupa(x,y), *Deleguj*(x,y), *Deleguj_B*(x,y), *Podmiot*(x), *Rola*(x), *Plik*(x), *Usuń*(x),
Przysnaj_rolę(x,y), *Zakres*(x, y,z), *Odgrywa*(x, y), *Dzień*(x), *Twórz_kopię*(x), *Dziedziczy*(x),
Zawiera_się(x,y,z), ...
- d) stałych logicznych (spójniki logiczne i kwantyfikatory):
 $\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \forall, \exists$
- e) operatorów deontycznych:
P, O, F,
odczytywanych w następujący sposób:
O ϕ - ϕ jest obowiązkowe,
P ϕ - ϕ jest dozwolone,
F ϕ - ϕ jest zabronione.
- f) nawiasów użytych jako znaków przystankowych.

Reguły formowania języka modelu *RMM*.

Formułą języka jest:

- a) formuła atomowa, tzn. n -argumentowy predykat z n -krotnym wystąpieniem w nim zmiennych indywiduowych i/lub stałych indywiduowych,
- b) negacja formuły,
- c) implikacja, koniunkcja lub alternatywa utworzona z dowolnych dwóch formuł,
- d) wyrażenie powstałe z poprzedzenia formuły kwantyfikatorem,
- e) jeśli ϕ jest formułą, to także **O** ϕ , **P** ϕ i **F** ϕ jest formułą,

Aksjomaty *RMM*.

Do aksjomatów *RMM* należą:

- a) wszystkie aksjomaty *FOL*,
- b) aksjomat zdaniowej logiki deontycznej:

$$D = O\varphi \rightarrow P\varphi$$

Reguły inferencyjne.

Do reguł inferencyjnych *RMM* należą wszystkie reguły z *FOL*.

W przypadku logik modalnych istotnym zagadnieniem jest precyzyjne zdefiniowanie semantyki tworzonego języka. Od niej to bowiem zależeć będzie interpretacja zapisywanych w danym języku wyrażeń, a w efekcie końcowym, zakres możliwości opisywania rzeczywistości przy wykorzystaniu danego formalizmu [Bat94], [Bor77], [Grz84], [Gum90], [Mar87].

Definicja 5.4. SEMANTYKA JĘZYKA *RMM*.

Niech model Kripkego dla logiki modelu *RMM* będzie opisany przez trójkę [Kri63]:

$$\langle K, w_0, v, \rangle$$

gdzie:

$K = \langle W, R, \rangle$ - jest strukturą Kripkego, gdzie:

- W - dowolny zbiór (tzw. światów możliwych),
- R - relacja binarna określająca istniejące powiązania pomiędzy światami możliwymi,

$w_0 \in W$ - jest wyróżnionym światem,

Z - jest zbiorem formuł języka *RMM*,

v - jest funkcją przyporządkowującą wartości logiczne zdaniom atomowym w światach $v: Z \times W \rightarrow \{T, F\}$ (wartości logiczne $T \equiv$ prawda, $F \equiv$ fałsz)

W takim przypadku, logiką modalną pierwszego rzędu o modalności **O** nazwiemy logikę $M = \langle F, C, \models_M \rangle$, gdzie:

C - jest pewną podklasą klasy modeli Kripkego,

$\models_M$ - relacja zdefiniowana w następujący sposób:

- $K, w, v, \models_m \varphi \Leftrightarrow v(\varphi, w) = T$,
- $K, w, v, \models_m O\varphi \Leftrightarrow$ dla każdego w' takiego, że $R(w, w')$ mamy $v(\varphi, w') = T$,
- $K, w, v, \models_m P\varphi \Leftrightarrow$ istnieje takie w' , że $R(w, w')$ mamy $v(\varphi, w') = T$

Przyjęty aksjomat $D = O\varphi \rightarrow P\varphi$ określa typ relacji R , która zachodzi pomiędzy światami możliwymi. Szersze omówienie typu tej relacji oraz konsekwencji z niej wypływającej dla modelowania sterowania dostępem zostaną omówione w dalszych częściach tego rozdziału.

5.3. Semantyka języka *RMM* w środowisku RBAC.

Semantyka języka modelu *RMM* oparta jest na propozycji semantyki zaprezentowanej w książce „Wnioskowanie w logikach nieklasycznych” (t. I i II) [BDRS95] [BDRS98], odwołującej się do semantyki Kripkego [Kri63]. Przyjęcie takiej definicji semantycznej pozwoli na odwoływanie się w dalszej części pracy do mechanizmu translacji semi-funkcyjnej [BDRS98], a przez to umożliwi wykorzystanie mechanizmów automatyzacji wnioskowania.

Definicja 5.5. Interpretacja elementów semantyki języka *RMM*.

Elementom struktury Kripkego nadaję następującą interpretację:

W - zbiór światów możliwych, będzie interpretowany jako zbiór możliwych stanów aktywności podmiotów w systemie. Zakładam, przy tym, że stan systemu da się jednoznacznie opisać poprzez określenie stanów dla wszystkich elementów aktywnych w systemie (podmiotów) oraz wartości zdefiniowanych w systemie zmiennych globalnych

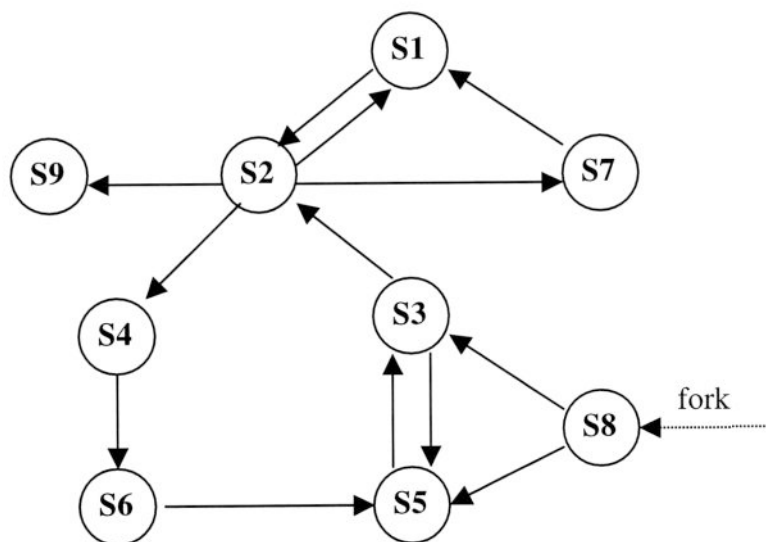
$\forall w \in W$ - każdy ze światów możliwych będzie interpretowany jako aktualny stan aktywności określonego podmiotu w systemie,

R - relacja wiążąca światy możliwe będzie interpretowana jako funkcja przejścia pomiędzy poszczególnymi stanami aktywności podmiotów.

Przykład 5.1. INTERPRETACJA SEMANTYKI ŚWIATÓW MOŻLIWYCH.

Rozważmy diagram stanów procesów i przejść między procesami w systemie Unix. Następująca lista zawiera pełny zbiór stanów procesu:

- S1. Proces wykonuje się w trybie użytkownika.
- S2. Proces wykonuje się w trybie jądra.
- S3. Nie wykonuje się, lecz jest gotowy do wykonania i czeka jedynie, aż zostanie wybrany przez jądro.
- S4. Śpi i przebywa w pamięci głównej.
- S5. Jest gotów do wykonania, lecz musi wcześniej być przesłany do pamięci głównej.
- S6. Śpi, a proces wymiany przesłał go do pamięci pomocniczej.
- S7. Wraca z trybu jądra do trybu użytkownika, lecz jądro wywłaszcza go i przełącza kontekst, by wybrać do wykonania inny proces.
- S8. Proces jest nowo utworzonym procesem i jest w stanie przejściowym.
- S9. Proces wykonał funkcję systemową *exit*, lecz jest w stanie *zombie*.



rys. 5.1. Diagram przejść międzystanowych procesu.

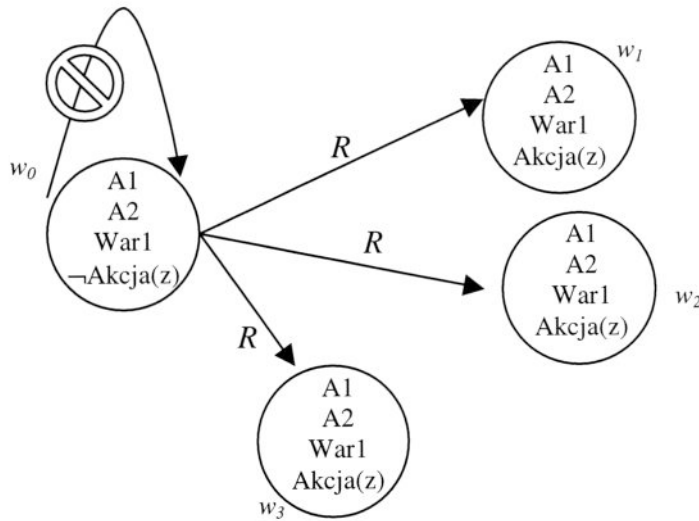
Zdefiniujmy zbiór światów możliwych dla procesu w systemie Unix:

$$W = \{w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8, w_9\}$$

Poza powyższym warunkiem nie nakładamy na relacje R żadnych innych dodatkowych wymagań takich jak np. żądanie aby była ona zwrotna, przechodnia, itp.

W odniesieniu do rozważanej rzeczywistej domeny światów możliwych (stany aktywności systemu) taka własność relacji R oznacza, że:

- dla każdego bieżącego stanu aktywności podmiotu/systemu istnieje co najmniej jeden możliwy stan następny. W szczególności może to być ten sam stan. Przejście w kolejnym kroku¹ przez podmiot/system do stanu będącego stanem bieżącym oznacza, że podmiot/system nie zmienił swojego stanu,
- brak wymogu zwrotności relacji (tzn. brak wymagania aby zachodziła relacja $\forall xR(x,x)$), powoduje że dla warunków typu $\text{War}_1 \rightarrow \mathbf{O}Akcja(z)$, możliwe jest przejście do stanu w którym spełnione jest jedynie wyrażenie War_1 , a wartość logiczna wyrażenia $Akcja(z)$ pozostaje wciąż fałszem. Dla relacji R z warunkiem zwrotności otrzymalibyśmy sprzeczny z intuicją wymóg jednoczesnej zmiany wartości logicznej War_1 i $Akcja(z)$ z F na T. (uzasadnienie tego powyższego spostrzeżenia otrzymujemy bezpośrednio z definicji operatora $\mathbf{O}$),



rys.5.2. Brak własności zwrotności relacji R .

W języku naturalnym powyżej rozważany warunek odczytujemy jako:

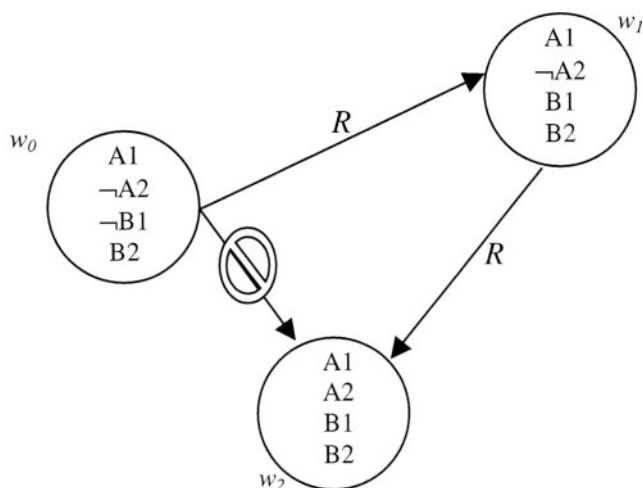
Jeżeli jest spełniony warunek War_1 , to podmiot jest zobowiązany do podjęcia akcji $Akcja(z)$.

Oczekujemy zatem, że zaistnienie War_1 , aktywuje dopiero konieczność $Akcja(z)$ i podmiot będzie musiał w następnym kroku doprowadzić do tego aby $Akcja(z)$ była spełniona. W przypadku zwrotności relacji R , interpretacja warunku musiała by być inne. Mianowicie mogła by ona być wyrażona w postaci następującego zdania:

¹ Pojęcie *kroku* nie jest przeze mnie formalizowane w tej pracy, gdyż istnieje stosunkowo duża swoboda w przyjęciu co będzie wyznacznikiem kolejnych kroków, a nie ma to znaczącego wpływu na poprawność przeprowadzanych rozumowań. *Krokiem* może to być w szczególności pojedynczy cykl zegara, zmiana stanu

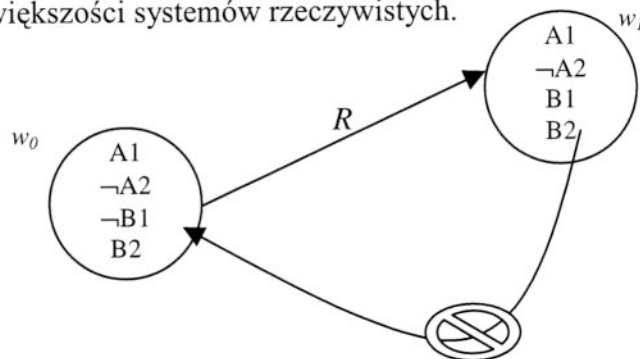
W kroku w którym zostanie spełniony warunek War_1 , podmiot musi zapewnić prawdziwość formuły $Akcja(z)$ oraz w każdym możliwym następnym kroku musi również zachodzić $Akcja(z)$.

- brak wymogu przechodniości ($\forall xyz(R(x,y) \wedge R(y,z)) \rightarrow R(x,z)$), otwiera konstruowany mechanizm na możliwość określania kolejności sekwencji wykonywanych operacji, tzn. daje możliwość wykluczenia przeskoków od stanu początkowego, bezpośrednio do stanu końcowego, bez możliwości weryfikowania etapów pośrednich. Własność ta jest szczególnie istotna w aplikacjach typu *work-flow*.



rys.5.3. Brak własności przechodniości relacji R .

- brak wymogu symetrii relacji ($\forall xy(R(x,y) \rightarrow R(y,x))$) umożliwia oddanie najczęściej spotykanego w rzeczywistości stanu, a mianowicie nieodwracalności pewnych operacji. Przy istnieniu symetrii, każdy podmiot, musiałby mieć zagwarantowaną możliwość do stanu, w którym przebywał w poprzednim kroku, a co jak wiadomo jest sprzeczne z charakterem większości systemów rzeczywistych.



rys.5.4. Brak własności symetrii relacji R .

aktywności dowolnego podmiotu w systemie, etc. Ważne jest jedynie aby przyjąć jednolite znaczenie *kroku* w

5.4. Zastosowanie modelu *RMM* do opisu *RBAC*.

W tej części rozdziału piątego zostaną przedstawione zasady wykorzystania zdefiniowanego wcześniej języka *RMM* w definiowaniu i zarządzaniu polityką sterowania dostępem opartą o role (*RBAC*). Definicja i elementy funkcjonalne rozważanej w tym i następnych rozdziałach polityki *RBAC* są zgodne z opisem w rozdziale czwartym.

5.4.1. Założenia wstępne.

Ogólna idea przedstawionego formalizmu dla systemu sterowania dostępem polega na wyrażeniu warunków dostępu oraz żądań dostępu w języku formalnym, a następnie sprawdzeniu czy zgłaszane żądanie należy do zbioru logicznych konsekwencji zbioru obowiązujących dany podmiot unormowań. Przy założeniu że zbiór konsekwencji przyjętych uregulowań jest tożsamy ze zbiorem praw dostępu w tradycyjnych rozwiązaniach, operacja ta będzie ze względu na bezpieczeństwo systemu tożsama ze sprawdzeniem czy danemu podmiotowi przysługuje prawo dostępu wymienione w zbiorze przynależnych mu praw [Kol99b]. Zaletą podejścia z wykorzystaniem mechanizmów logiki formalnej do problemu definiowania zbioru uprawnień dla aktywnych podmiotów jest to, że można zmniejszyć liczbę warunków koniecznych do zapamiętania przez system, a które są niezbędne do jednoznacznego określenia domeny dozwolonych akcji. Możemy się zatem ograniczyć do określenia jedynie zbioru X ogólnych reguł rządzących aktywnością danego podmiotu w systemie, bez konieczności przechowywania definicji wszystkich możliwych wariantów jego postępowania (tożsamego ze zbiorem konsekwencji zbioru X , czyli zbioru $Y=Cn(X)$). Wynika to z następującego twierdzenia [Bat94]:

$$X \subset Cn(X)$$

Które stwierdza że zbiór X zawiera się w zbiorze konsekwencji tego zbioru, czyli:

$$card(X) \leq card(Cn(X))$$

W związku z powyższym pierwsze założenie przyjmuje postać:

Założenie 1.

Jeżeli Reg jest zbiorem formuł wyznaczających podstawowe prawa dostępu dla danego podmiotu, to podmiot ten jest uprawniony do wykonania każdej akcji określonej przez formułę $Action$, która spełnia warunek:

$$Action \in Cn(Reg)$$

Zgodnie z treścią *założenia 1.*, aby móc skutecznie sterować dostępem do zasobów systemu, konieczne jest aby system posiadał możliwość jednoznacznego określania zbioru konsekwencji logicznych zbioru uregulowań [Lee67]. W rzeczywistości nie będziemy potrzebowali znajdować każdorazowo pełnego zbioru konsekwencji logicznych dla danego zbioru uregulowań. Wystarczy, że będziemy w stanie w jednoznaczny sposób udzielić odpowiedzi, czy kierowane do systemu żądanie dostępu należy do zbioru konsekwencji obowiązującego zbioru norm.

Ponieważ logika pierwszego rzędu (a przez to i jej modalne rozszerzenia) jest nierozstrzygalna, to zgodnie z *twierdzeniem Churcha* nie istnieje mechaniczna metoda sprawdzania czy dana formuła języka tej teorii jest czy nie jest dowodliwa (tzn. czy należy do zbioru konsekwencji przyjętych aksjomatów) [Mar87][BDRS95]. Z drugiej jednak strony mamy twierdzenie Gödla, które mówi, że zbiór tautologii klasycznej logiki pierwszego rzędu jest częściowo rozstrzygalny oraz w innym twierdzeniu udowodnionym przez niego znajdujemy, że dla każdego częściowo rozstrzygalnego zbioru X istnieje całkowity algorytm $\varphi: X \rightarrow F_I$ o tej własności, że dla dowolnego x mamy [God67]:

$$x \in X \Leftrightarrow \models \varphi(x)$$

W szczególności wnioskowanie w każdej logice o częściowo rozstrzygalnym zbiorze twierdzeń można efektywnie sprowadzić do wnioskowania w logice klasycznej.

Proces automatyzacji wnioskowania dla *FOL* jest dobrze znany i istnieje wiele algorytmów pozwalających na efektywne rozstrzyganie o prawdziwości formuł tej logiki. W opracowywanym mechanizmie sterowania dostępem skorzystano z możliwości translacji języka *RMM* do postaci *FOL*.

W tym celu wykorzystano algorytmy translacji semi-funkcyjnej formuł, schematów i reguł modalnych podane w [BDRS95]². Algorytmy te pozwalają na tłumaczenie szerokiej klasy logik modalnych na logikę klasyczną przy zachowaniu semantyki logiki pierwotnej. Umożliwia to późniejsze zastosowanie mechanizmów automatyzacji procesu wnioskowania opracowanych dla logik klasycznych, np. PROLOGu [PoS70], [PSu96], [RL91].

Program w PROLOGu jest ciągiem klauzul Horna. Wykonanie programu logicznego polega na odpowiedzi na zadanie pytanie, które może być koniunkcją literałów pozytywnych. Sens pytania przy tym jest następujący:

Czy formuła stanowiąca pytanie jest logiczną konsekwencją treści programu?

Odpowiedź w najprostszym przypadku jaką otrzymamy w wyniku działania programu jest TAK lub NIE.

Zauważmy, że zasada działania programu w PROLOGu odpowiada określonej przez nas zasadzie poszukiwania odpowiedzi na pytanie czy dana akcja jest, czy też nie jest dozwolona w świetle obowiązujących uregulowań dotyczących sterowania dostępem. Dlatego też zasadę

² Ich treść zamieszczono w dodatku B niniejszej pracy.

rezolucji, która jest głównym mechanizmem automatyzacji wnioskowania w PROLOGu uczynimy podstawowym mechanizmem wnioskowania w konstruowanym mechanizmie sterowania dostępem. W tym celu należy określić jakie są ograniczenia narzucone na treść programów logicznych w PROLOGU, gdyż będą one również obowiązywać dla opracowywanego mechanizmu automatyzacji wnioskowania w języku *RMM*.

Klauzule Horna będące elementami programów w PROLOGU mają następującą ogólną postać:

$$\phi \vee \neg\phi_1 \vee \neg\phi_2 \vee \neg\phi_3 \vee \neg\phi_4 \vee \dots \vee \neg\phi_n$$

gdzie $\phi, \phi_1, \phi_2, \dots$ literały.

Oznacza to, że tylko jeden literał w całej klauzuli może być pozytywny. Formuła ta jest inną formą zapisu implikacji następującej postaci:

$$\phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4 \wedge \dots \wedge \phi_n \rightarrow \phi$$

Zgodnie z przyjętą konwencją klauzule zapisuje się w zgodzie z poniższym sposobem:

$$\phi \Leftarrow \phi_1, \phi_2, \phi_3, \phi_4, \dots, \phi_n$$

Konieczne jest zatem sprawdzenie, czy wejściowa formuła będąca wynikiem tłumaczenia formuły logiki modalnej na klasyczną jest w postaci klauzulowej. Dla logik serialnych³ istnieje proste kryterium pozwalające rozstrzygnąć czy dana reguła modalna po przetłumaczeniu jest w oczekiwanej przez nas postaci klauzulowej [BDRS98].

Twierdzenie 5.1. O MOŻLIWOŚCI TRANSLACJI FORMUŁ JĘZYKA RMM DO POSTACI KONIUNKCYJNEJ KLAUZUL HORNA (KKH)

$T_{sf}(\phi, w)$ jest koniunkcją klauzul Horna wtedy i tylko wtedy, gdy formuła uzyskana z ϕ poprzez usunięcie operatorów modalnych jest koniunkcją klauzul Horna,

Gdzie: ϕ - formuła języka RMM, $T_{sf}()$ - translacja semi-funkcyjna (patrz Dodatek B.).

³ patrz Lemat 5.1.

Dowód

Dowód powyższego twierdzenia można przeprowadzić przy użyciu indukcji strukturalnej.

Niech ϕ i ψ będą formułami RMM.

Krok 1.

Niech ϕ będzie literałem. Otrzymujemy wtedy:

- ϕ - jest w postaci koniunkcyjnej klauzul Horna (kkH),
- $\mathbf{O}\phi$ - po opuszczeniu operatora modalnego otrzymujemy ϕ , jest w postaci kkH,
- $\mathbf{P}\phi$ - tak samo jak w przypadku operatora $\mathbf{O}$

Zgodnie z definicją translacji T_{sf} otrzymujemy:

- a) $T_{sf}(\phi, x) = P(x)$
Jest w postaci kkH
- b) $T_{sf}(\mathbf{O}\phi, x) = \forall y[R(x, y) \rightarrow T_{sf}(\phi, y)] = \forall y[\neg R(x, y) \vee T_{sf}(\phi, y)] = \forall y[\neg R(x, y) \vee P(y)]$
Jest w postaci kkH
- c) $T_{sf}(\mathbf{P}\phi, x) = \exists f T_{sf}(\phi, f(x)) = \exists f P(f(x))$
Jest w postaci kkH

Krok 2.

Założenia:

- ψ - formuła, której $T_{sf}(\psi, x)$ jest w postaci kkH $\Leftrightarrow \psi$ po odrzuceniu wszystkich operatorów modalnych jest w postaci kkH,
- ϕ - dowolny literał,

Pokażemy, że formuły złożone $\mathbf{O}\phi \rightarrow \psi$, $\psi \rightarrow \mathbf{O}\phi$, $\psi \wedge \mathbf{O}\phi$, $\psi \vee \mathbf{O}\phi$, są w postaci kkH $\Leftrightarrow$ gdy formuły $\phi \rightarrow \psi$, $\psi \rightarrow \phi$, $\psi \wedge \phi$, $\psi \vee \phi$, są w postaci kkH.

- $\psi \vee \mathbf{O}\phi$, a po odrzuceniu operatora modalnego otrzymujemy $\psi \vee \phi$,
 $T_{sf}(\psi \vee \mathbf{O}\phi, x) = T_{sf}(\psi, x) \vee T_{sf}(\mathbf{O}\phi, x) = T_{sf}(\psi, x) \vee \forall y [\neg R(x, y) \vee P(y)]$

- a) $\psi \vee \phi$ - jest w postaci kkH

Ponieważ otrzymana w wyniku translacji semi-funkcyjnej formuła jest klauzulą, w której nie występuje więcej literałów pozytywnych niż w formule $\psi \vee \phi$, to $T_{sf}(\psi \vee \mathbf{O}\phi, x)$ jest w postaci kkH,

- b) $\psi \vee \phi$ - nie jest w postaci kkH

Ponieważ $\psi \vee \phi$ nie jest w postaci kkH, to również klauzula będąca rozwinięciem $T_{sf}(\psi \vee \mathbf{O}\phi, x)$ nie będzie w postaci kkH.

- $\psi \wedge \mathbf{O}\phi$, a po odrzuceniu operatora modalnego otrzymujemy $\psi \wedge \phi$,

$$T_{sf}(\psi \wedge \mathbf{O}\phi, x) = T_{sf}(\psi, x) \wedge T_{sf}(\mathbf{O}\phi, x) = T_{sf}(\psi, x) \wedge \forall y [\neg R(x, y) \vee P(y)]$$

- a) $\psi \wedge \phi$ - jest w postaci kkH

Ponieważ otrzymana w wyniku translacji semi-funkcyjnej formuła jest koniunkcją klauzul, której obydwa składniki są w postaci kkH, to również $T_{sf}(\psi \wedge \mathbf{O}\phi, x)$ jest w postaci kkH,

- b) $\psi \wedge \phi$ - nie jest w postaci kkH

Ponieważ $\psi \wedge \phi$ nie jest w postaci kkH, oznacza to, że ψ nie jest w postaci kkH (gdyż ϕ jest literałem i jako taki nie może być przyczyną niezgodności z postacią kkH). W związku z tym, z założenia indukcyjnego $T_{sf}(\psi, x)$ nie jest w postaci kkH, a z tego wynika, że $T_{sf}(\psi \wedge \mathbf{O}\phi, x)$ jest w postaci kkH.

- $\psi \rightarrow \mathbf{O}\phi$, a po odrzuceniu operatora modalnego otrzymujemy $\psi \rightarrow \phi$,

$$\begin{aligned} T_{sf}(\psi \rightarrow \mathbf{O}\phi, x) &= T_{sf}(\psi, x) \rightarrow T_{sf}(\mathbf{O}\phi, x) = T_{sf}(\psi, x) \rightarrow \forall y [\neg R(x, y) \vee P(y)] = \\ &= \neg T_{sf}(\psi, x) \vee \forall y [\neg R(x, y) \vee P(y)] \end{aligned}$$

- a) $\psi \rightarrow \phi$ ($\neg\psi \vee \phi$) - jest w postaci kkH

- b) $\psi \rightarrow \phi$ ($\neg\psi \vee \phi$) - nie jest w postaci kkH

ad. a) i b)

Korzystając z równoważnej postaci koniunkcji w postaci alternatywy, rozumowanie można sprowadzić do tego jakie miało miejsce przy dowodzie twierdzenia dla alternatywy.

- $\mathbf{O}\phi \rightarrow \psi$, a po odrzuceniu operatora modalnego otrzymujemy $\phi \rightarrow \psi$,

$$\begin{aligned} T_{sf}(\mathbf{O}\phi \rightarrow \psi, x) &= T_{sf}(\neg \mathbf{O}\phi \vee \psi, x) = T_{sf}(\mathbf{P} \neg \phi \vee \psi, x) = T_{sf}(\mathbf{P} \neg \phi, x) \vee T_{sf}(\psi, x) = \\ &= \exists f T_{sf}(\neg \phi, f(x)) \vee T_{sf}(\psi, x) = \exists f \neg T_{sf}(\phi, f(x)) \vee T_{sf}(\psi, x) = \exists f \neg P(f(x)) \vee T_{sf}(\psi, x) \end{aligned}$$

- a) $\phi \rightarrow \psi$ ($\neg\phi \vee \psi$) - jest w postaci kkH

- b) $\phi \rightarrow \psi$ ($\neg\phi \vee \psi$) - nie jest w postaci kkH

ad. a) i b)

Formuła $\neg\phi\vee\psi$ jest w postaci kkH $\Leftrightarrow$ jeżeli ψ jest w postaci kkH. Ponieważ w wyniku translacji semi-funkcyjnej formuły $\mathbf{O}\phi\rightarrow\psi$ otrzymaliśmy klauzulę składającą się z $T_{sf}(\psi,x)$ i negacji $P(f(x))$, to również $T_{sf}(\mathbf{O}\phi\rightarrow\psi,x)$ jest w postaci kkH dla ψ w postaci kkH i nie jest w postaci kkH dla ψ w nie będącego w postaci kkH.

Dowód prawdziwości twierdzenia dla dualnego operatora modalnego **P** wygląda analogicznie do tego, który został przedstawiony powyżej.

q.e.d.

W oparciu o powyższe twierdzenie możemy zdefiniować algorytm, który pozwoli na ostateczne zweryfikowanie czy dana formuła języka *RMM* jest w postaci kkH.

Algorytm 5.1. SPRAWDZENIE CZY BADANA FORMUŁA JĘZYKA *RMM* JEST W POSTACI KKH.

Dane: X - dowolna formuła logiki *RMM*

Wyjście: TAK - jeżeli dana formuła jest w postaci kkH,

NIE - jeżeli nie jest w postaci kkH.

Procedura:

1. Opuść wszystkie operatory modalne,
2. Sprowadź X do przedrostkowej postaci normalnej *PNF*,
3. Uzyskaną formułę sprowadź do postaci Skolema,
4. Wyjście kroku 4. działania procedury sprowadź do koniunkcyjnej postaci normalnej *CNF*,
5. **While** istnieje kolejny element koniunkcji formuły w postaci *CNF*
6. **if** w danym składniku występuje więcej niż jeden literal pozytywny
 then return NIE
 return TAK
7. **end.**

Twierdzenie 5.2. O MOŻLIWOŚCI SPROWADZENIA REGUŁY JĘZYKA *RML* DO POSTACI KKH.

Formuły języka *RML* jest w postaci kkH, wtedy gdy rezultatem zwracanym przez *Algorytm 5.1.* na którego wejściu podano badaną formułę, jest odpowiedź TAK

Dowód

Korzystając z Twierdzenia 5.1. oraz z definicji postaci *kkH*, własności przekształceń *CNF*, *PNF*, *skolemizacji*, otrzymujemy tezę twierdzenia.

q.e.d.

W powyżej przedstawiony sposób uzyskujemy łatwy sposób weryfikowania czy formułowane przez nas warunki są w postaci *kkH*, a dzięki temu czy możemy je wykorzystać w procesie automatyzacji wnioskowania.

Jeżeli badana przez nas formuła nie znajduje się w postaci *kkH*, to konieczne jest przeformułowanie jej, lub dodanie nowych predykatów (np. *Nie_jest_właścicielem(x)* zamiast $\neg \text{Jest_właścicielem}(x)$), tak aby sprowadzić formułę do postaci *kkH*. Operacja taka umożliwi proces analizy logicznej na drodze automatycznego wnioskowania, choć niewątpliwie może również przyczynić się do zmniejszenia przejrzystości zapisu w poszczególnych przypadkach. Nie powinno to niemniej jednak wpływać negatywnie na efektywność obliczeniową systemu sterowania. Wynika to z faktu, iż nie potrzebujemy tworzyć nowych funkcji obliczających wartość nowoutworzonego predykatu, a wystarczy jedynie dla otrzymania jego wartości, każdorazowo zanegować wynik jaki otrzymujemy przy wywołaniu „starego” predykatu przy odpowiednim podstawieniu wartości zmiennych i stałych.

W związku z powyższym formułujemy następujące założenie pomocnicze na drodze do stworzenia formalnego modelu polityki *RBAC*.

Założenie 2.

Definiowane przez nas w logice *RML* warunki, uprawnienia, etc., powinny być przedstawione w postaci koniunkcyjnej klauzur Horna.

Przyjęcie dwóch opisanych w tym punkcie założeń nie ma negatywnego wpływu na siłę ekspresji, a przez to i na możliwości praktycznego zastosowania, konstruowanego mechanizmu sterowania dostępem.

Przykład 5.2. RÓŻNICA POMIĘDZY KLASYCZNYM A MODALNYM PODEJŚCIEM DO STEROWANIA DOSTĘPEM.

Dzięki wykorzystaniu w modelu *RMM* logiki deontycznej wśród formuł definiujących prawa dostępu dla danej roli mogą pojawiać się również literały postaci: **$P\alpha$** , **$O\beta$** , **$F\delta$** . Literały te umożliwiają formalny opis faktu, że np. dana akcja jest dozwolona (**$P\alpha$**), konieczna (**$O\beta$**) czy też zabroniona (**$F\delta$**).⁴

⁴ W sposób ogólny możliwości logiki deontycznej zostały opisane w punkcie 5.1. Logika formalna i 5.1.1. Normatywny aspekt sterowania dostępem.

Ogólna postać formuł składających się na definicję roli polityki sterowania dostępem poprzez role ma postać (formuły są w postaci kKH):

$$War_1 \wedge War_2 \wedge \dots \wedge War_n \rightarrow Akcja$$

W związku z tym, formuły te mogą być zapisywane zgodnie z poniższą konwencją zapisu klauzul:

$$Akcja \Leftarrow War_1, War_2, \dots, War_n$$

Rozpatrzmy sytuację, gdy ciąg warunków wstępnych jest pusty tzn. formułę postaci:

$$True \rightarrow Akcja,$$

którą to nazywamy asercją i w konwencji zapisu klauzul zapisujemy ją jak poniżej:

$$Akcja \Leftarrow$$

Pokażemy różnicę pomiędzy sytuacją kiedy:

a) $Akcja \in \{ \mathbf{P}\alpha, \mathbf{O}\alpha, \mathbf{F}\alpha \},$

a sytuacją gdy:

b) $Akcja \in \{ \alpha, \neg\alpha \}.$

To znaczy pokażemy w sposób formalny zalety wynikające z możliwości stosowania operatorów deontycznych.

ad. a)

Z definicji translacji semi-funkcyjnej otrzymujemy, że:

	Wynik translacji	Postać klauzulowa
$T_{sf}(\mathbf{P}\alpha, x)$	$\exists f P(f(x))$	$P(g(x)) \Leftarrow$
$T_{sf}(\mathbf{O}\alpha, x)$	$\forall y [\neg R(x, y) \vee P(y)]$	$P(y) \Leftarrow R(x, y)$
$T_{sf}(\mathbf{F}\alpha, x)$	$\forall y [\neg R(x, y) \vee \neg P(y)]$	$\Leftarrow P(y), R(x, y)$
$T_{sf}(\alpha, x)$	$P(x)$	$P(x) \Leftarrow$

Gdzie:

$g(x)$ - konkretna funkcja ze zbioru wszystkich funkcji spełniających relację dostępności światów (stanów),

x, y - zmienne reprezentujące światy możliwe (stany) .

Rozpatrujemy trzy przypadki.

- 1) Zbiór reguł Reg określający prawa dostępu podmiotów do zasobów systemu jest zdefiniowany jak poniżej

$$Reg = \{ \mathbf{P}\alpha \}$$

Badamy możliwość wykonania przez podmiot akcji α .

Wykorzystujemy mechanizm automatyzacji wnioskowania oparty na metodzie rezolucji.

Zbiorem wejściowym reguł będzie zatem:

- $P(g(x)) \Leftarrow$ - translacja formuły $\mathbf{P}\alpha$
- $R(x, f(x)) \Leftarrow$ - klauzula dodawana każdorazowo w celu eliminacji możliwości zapętlenia się programu logicznego [BRDS98]

zaś pytanie ma następującą postać:

- $\Leftarrow P(x_0)$ - translacja formuły α

Przy takiej treści programu logicznego i dla takiego pytania otrzymujemy następujące odpowiedzi:

- $\Leftarrow$ - dla x_0 za które można podstawić $g(x)$, tzn. dla takich stanów, które są dostępne z x_0 w funkcji $g(x)$,

Oznacza to, że żądanie dostępu α jest zgodne ze zbiorem praw dostępu $\mathbf{P}\alpha$.

- $P(g(x)) \Leftarrow$ - dla x_0 za które nie można podstawić $g(x)$, tzn. dla takich stanów,
- $R(x, f(x)) \Leftarrow$ - które nie są dostępne z x_0 w funkcji $g(x)$,
- $\Leftarrow P(x_0)$

Oznacza to, że żądanie dostępu α nie jest zgodne ze zbiorem praw dostępu $\mathbf{P}\alpha$.

W związku z powyższym otrzymaliśmy, że akcja α jest dozwolona w stanie systemu x_0 , ale nie jest ona obowiązkowa (można przejść do kolejnego stanu nie wykonując α). Mechanizm kontroli dostępu zezwoli na wykonanie akcji α w przypadku zgłoszenia takiego żądania przez któryś z podmiotów.

- 2) Zbiór reguł Reg określający prawa dostępu podmiotów do zasobów systemu jest zdefiniowany jak poniżej

$$Reg = \{\mathbf{O}\alpha\}$$

Badamy możliwość wykonania przez podmiot akcji α .

Wykorzystujemy mechanizm automatyzacji wnioskowania oparty na metodzie rezolucji.

Zbiorem wejściowym reguł będzie zatem:

- $P(y) \Leftarrow R(x, y)$ - formuła uzyskana po translacji formuły $\mathbf{O}\alpha$
- $R(x, f(x)) \Leftarrow$ - klauzula dodawana każdorazowo w celu eliminacji możliwości zapętlenia się programu logicznego [BRDS98]

pytanie ma następującą postać:

$\Leftarrow P(x_0)$ - translacja formuły α

Przy takiej treści programu logicznego i dla takiego pytania otrzymujemy następującą odpowiedź:

$\Leftarrow$ - dla każdego stanu x_0 ,

Oznacza to, że żądanie dostępu α jest zgodne ze zbiorem praw dostępu $O\alpha$.

W związku z powyższym otrzymaliśmy, że akcja α jest konieczna w stanie systemu x_0 (nie można przejść do kolejnego stanu nie wykonując α).

Mechanizm kontroli dostępu zezwoli na wykonanie akcji α , w przypadku zgłoszenia takiego żądania przez któryś z podmiotów. Wszelkie próby przejścia do kolejnego stanu bez wykonania akcji α , system powinien sygnalizować jako niezgodne z polityką.

3) Zbiór reguł Reg określający prawa dostępu podmiotów do zasobów systemu jest zdefiniowany jak poniżej

$$Reg = \{F\alpha\}$$

Badamy możliwość wykonania przez podmiot akcji α .

Wykorzystujemy mechanizm automatyzacji wnioskowania oparty na metodzie rezolucji.

Zbiorem wejściowym reguł będzie zatem:

$\Leftarrow P(y), R(x,y)$ - translacja formuły $F\alpha$

$R(x,f(x)) \Leftarrow$ - klauzula dodawana każdorazowo w celu eliminacji możliwości zapętlenia się programu logicznego [BRDS98]

zaś pytanie ma następującą postać:

$\Leftarrow P(x_0)$ - translacja formuły α

Przy takiej treści programu logicznego i dla takiego pytania otrzymujemy następującą odpowiedź:

$\Leftarrow P(x_0)$ - dla każdego stanu x_0 ,

Oznacza to, że żądanie dostępu α nie jest zgodne ze zbiorem praw dostępu $F\alpha$.

W związku z powyższym otrzymaliśmy, że akcja α jest zabroniona w stanie systemu x_0 (nie można przejść do kolejnego stanu wykonując α).

Mechanizm kontroli dostępu nie zezwoli na wykonanie akcji α , w przypadku zgłoszenia takiego żądania przez któryś z podmiotów. Wszelkie próby przejścia do kolejnego stanu poprzez wykonanie akcji α , system powinien sygnalizować jako niezgodne z polityką.

ad. b)

W klasycznym podejściu mamy tylko dwie możliwości:

$$Reg = \{\alpha\} \text{ lub } Reg = \{\neg\alpha\}$$

1)

Dla $Reg = \{\alpha\}$ akceptowane będą żądania wykonania akcji α , gdyż treść programu logicznego w tym przypadku będzie składała się wyłącznie z reguły: $\alpha \Leftarrow$.

Rezultat ten jest zbliżony do otrzymanego dla przypadku $Reg = \{\mathbf{O}\alpha\}$ w podejściu modalnym.

2)

Dla $Reg = \{\neg\alpha\}$ odrzucane będą żądania wykonania akcji α , gdyż treść programu logicznego w tym przypadku będzie składała się wyłącznie z reguły: $\Leftarrow\alpha$.

Rezultat ten jest zbliżony do otrzymanego dla przypadku $Reg = \{\mathbf{F}\alpha\}$ w podejściu modalnym.

Podsumowując, ujęcie klasyczne (tzn. bez operatorów modalnych) charakteryzują następujące wady w porównaniu do podejścia z wykorzystaniem logiki deontycznej:

- trudno jest nadać właściwą i intuicyjną interpretację formułom wchodzącym w skład zbioru określającego dopuszczalne akcje dla podmiotów,

Na przykład, jaką interpretację nadać $Reg = \{\alpha\}$?

α jest dozwolone,

czy też

α jest konieczne.

- brak jest możliwości reprezentowania jednoczesnego reprezentowania pojęcia konieczności, zezwolenia i zakazu wykonywania akcji dostępu.

5.5. Podsumowanie.

W rozdziale piątym przedstawiono definicję modelu *RMM* (RBAC Modal Model) sterowania i zarządzania dostępem do systemu informatycznego. Model ten umożliwia zapis polityki dostępu oraz sterowanie i zarządzanie dostępem w sposób zgodny z podstawowymi założeniami polityki RBAC.

Elementami modelu jest język modalny z deontycznymi operatorami modalnymi **O**, **P**, **F**, o znaczeniu odpowiednio: *jest obowiązkowe, że (O)*, *jest dozwolone, że (P)*, *jest zakazane, że (F)* oraz zasady wykorzystania tego języka na potrzeby sterowania i zarządzania dostępem poprzez role.

Udowodnione zostały twierdzenia o możliwości translacji formuł języka *RMM* do postaci koniunkcji klauzul Horna (kkH) i o możliwości sprowadzenia reguły języka *RML* do postaci kkH oraz opracowano algorytm sprawdzania czy badana formuła języka *RMM* jest w postaci kkH.

6. Mechanizm sterowania dostępem.

W tym rozdziale zostanie zdefiniowany algorytm pozwalający systemowi sterowania dostępem na sprawdzanie czy żądanie wykonania danej operacji zgłaszane przez podmiot, jest zgodne z przysługującymi mu prawami, wynikającymi z roli lub ról, które odgrywa on w systemie. Dzięki temu będzie możliwe wypracowanie przez system sterowania jednoznacznej decyzji o odrzuceniu bądź zezwoleniu na wykonanie tejże operacji.

6.1. Analiza spójności żądań podmiotów z przyjętą polityką.

Żądanie wykonania operacji generowane przez podmiot jest opisane w zdefiniowanym wcześniej języku formalnym oraz założmy dla uproszczenia rozważań, że jest opisane w formie implikacji (konieczne jest jedynie tylko to aby było ono w postaci dającej sprowadzić się do postaci kkH)⁵ w postaci takiej jak np.

$$\text{Podmiot}(\text{Bartek}) \wedge \text{Rola}(\text{Admin}) \wedge \text{Plik}(\text{Dane_osobowe}) \rightarrow \text{PUsun}(\text{Dane_osobowe})$$

Punktem wyjściowym do zbadania czy system powinien, czy też nie powinien zezwolić na wykonanie danej operacji, jest prawdziwość następujących założeń o stanie systemu:

- możliwe jest zidentyfikowanie podmiotu wysuwającego żądanie wykonania operacji,
- możliwe jest zidentyfikowanie zbioru ról jakie są skojarzone z danym podmiotem oraz zbioru formuł określających pełen zakres możliwości podmiotu (tzn. system jest w stanie odtworzyć wszelkie bezpośrednio przyznane uprawnienia jak i te przysługujące podmiotowi na drodze dziedziczenia i delegacji uprawnień),
- zbiór uprawnień przysługujący podmiotowi nie jest sprzeczny.

Jeżeli którekolwiek z powyższych założeń nie może być spełnione, wszelkie decyzje podejmowane przez system mogą być niezgodne z pierwotnie przyjętą polityką sterowania dostępem. Dla zapewnienia bezpieczeństwa, system w chwili wykrycia niezgodności swojego stanu z założeniami wstępnymi, powinien odmówić wykonania zgłaszanego żądania i zasygnalizować awarię [BC93], [Cup94].

Jeżeli założenia są spełnione, to proponowany przeze mnie mechanizm sterowania dostępem jest w stanie określić zgodność żądania z przyjętą polityką bezpieczeństwa na drodze realizacji poniższego algorytmu.

⁵ patrz założenie 2,

Algorytm 6.1. ANALIZA SPÓJNOŚCI ŻAŁAŁ PODMIOTÓW Z PRZYJĘTĄ POLITYKĄ.

Dane: f - dowolna formuła *RMM* określająca żądanie dostępu,

Wyjście: TAK - jeżeli podmiot jest uprawniony do wykonania danej akcji,

NIE - jeżeli podmiot nie jest uprawniony do wykonania danej akcji,

Procedura:

1. Identyfikacja kontekstu wykonania zamierzonej operacji.

Do kontekstu należy przede wszystkim nazwa podmiotu zgłaszającego żądanie, na podstawie której system określa role powiązane z tym podmiotem, z przez to i zbiór obowiązujących go norm (*Reg*). Elementami kontekstu może być również nazwa, typ obiektu, którego wykonywana operacja ma dotyczyć, czas zgłoszenia żądania, itp. w zależności od typu operacji.

2. Wyznaczenie treści programu logicznego na podstawie zbioru norm (*Reg*) obowiązujących dany podmiot.

3. Do programu logicznego kierujemy pytanie, które jest zanegowaną formułą opisującą zgłaszane żądanie (f) dostępu do zasobów systemu przez podmiot.

4. Uruchamiamy program logiczny, który generuje odpowiedź na zadane pytanie wykorzystując mechanizm automatyzacji wnioskowania w oparciu o regułę rezolucji⁶

5. Decyzja w oparciu o odpowiedź programu logicznego.

if program logiczny w oparciu o *Reg* i zapytanie f wygenerował odpowiedź TAK

return TAK

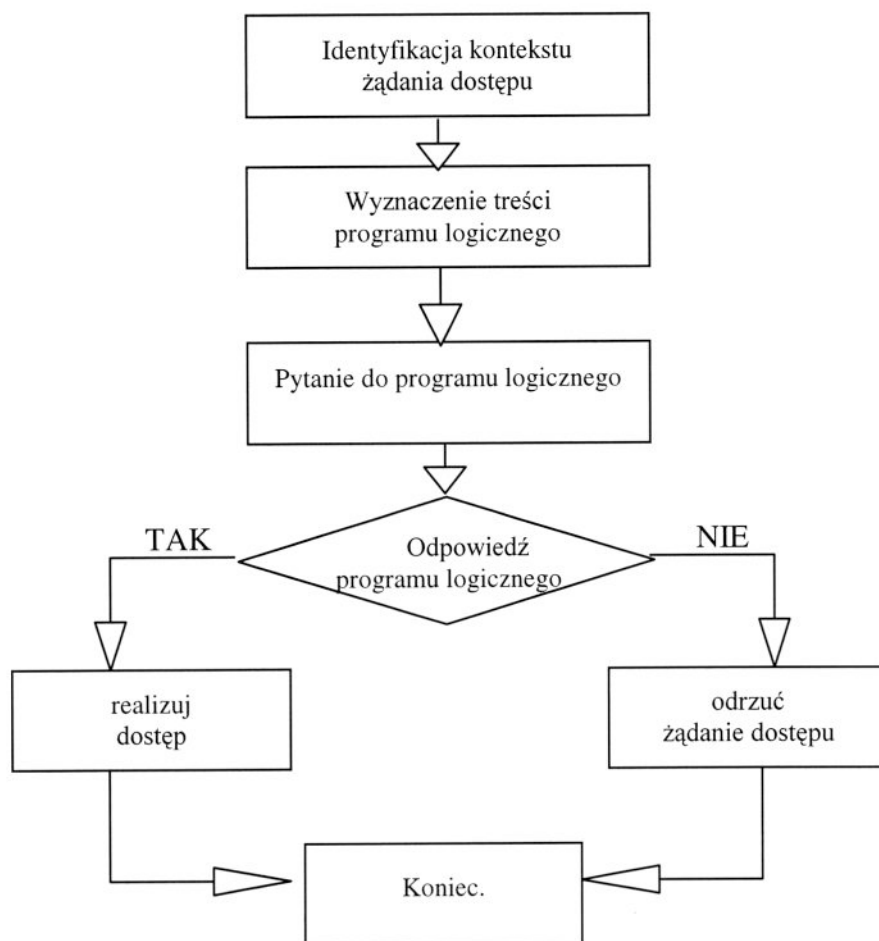
- zezwolenie na wykonanie operacji, operacja jest logiczną konsekwencją zbioru norm obowiązujących dany podmiot.

else return NIE

- odrzucenie żądania wykonania operacji, operacja jest sprzeczna z normami obowiązującymi podmiot.

6. Koniec.

⁶ patrz dodatek B



rys. 6.1. Schemat blokowy algorytmu 6.1.

Aby algorytm mógł być zaakceptowany jako podstawa systemu sterowania musi on gwarantować, że wszystkie żądania niezgodne z przyjętą polityką będą przez niego odrzucone.

Twierdzenie 6.1. O DOPUSZCZENIU DO WYKONANIA PRZEZ PODMIOT OPERACJI DOSTĘPU.

Żądanie dostępu zgłaszane przez podmiot (analizowane zgodnie z *Algorytmem 6.1.*) zostanie zaakceptowane wtedy i tylko wtedy, gdy jest ono zgodne z przyjętą polityką kontroli dostępu i zapisaną przy pomocy formuł języka kontroli dostępu o zdefiniowanych uprzednio własnościach.

Dowód

Zgodnie z przyjętym założeniem, operacje dozwolone dla danego podmiotu wyznaczone są poprzez zbiór $X = Cn(Reg)$, gdzie Reg jest spójnym zbiorem norm zawartych w aktywnych i skojarzonych z danym podmiotem rolach. Algorytm działania programu logicznego (krok 4. algorytmu 6.1.) polega na generowaniu rezolwent zbioru klauzul $Reg \cup \{\neg\alpha\}$, gdzie α jest klauzulą wejściową, opisującą zamierzoną przez podmiot akcję. W wyniku działania tego algorytmu otrzymujemy odpowiedź TAK wtedy i tylko wtedy, gdy $\alpha \in Cn(Reg \cup \{\neg\alpha\})$. Ponieważ wiadomo również, że $\alpha \in Cn(Reg)$ iff $\alpha \in Cn(Reg \cup \{\neg\alpha\})$ [Kul99], otrzymujemy, że algorytm 6.1. zezwala na wykonanie operacji wtedy i tylko wtedy, gdy operacja ta należy do zbioru logicznych konsekwencji obowiązującego danego podmiotu zbioru norm, tzn. gdy jest zgodna z przyjętą polityką bezpieczeństwa.

q.e.d.

Przykład 6.1. WERYFIKACJA ZGODNOŚCI ŻAŁAŃ DOSTĘPU Z PRZYJĘTĄ POLITYKĄ.

Podmiot *Podmiot_1* posiada określony przez zbiór Reg zbiór uprawnień. Chcemy sprawdzić czy system sterowania dopuści do wykonania dwóch następujących operacji $Czytaj(Podmiot_1, Plik_a)$ i $Czytaj(Podmiot_1, Plik_b)$.

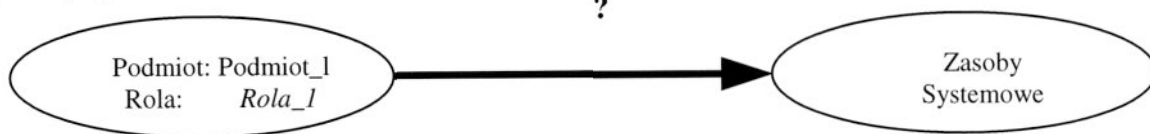
Dane:

- Podmiot *Podmiot_1* oraz przypisana mu rola *Rola_1*.
- *Rola_1* określona przez reguły R1 i R2:
 - R1. $\forall pd \forall pl \text{ Zakres}(pl, \text{Katalog_A}, \text{Katalog_B}) \wedge$
 $\wedge \text{Odgrywa}(pd, \text{Pracownik}) \rightarrow \mathbf{P} \text{Czytaj}(pd, pl)$
 - R2. $\forall pd \text{ Stanowisko}(pd, \text{Admin}) \rightarrow \neg \mathbf{P} \text{Przyznaj_role}(pd, \text{Pracownik})$
- Polityka określona poprzez wartości literalów:
 - $\text{Zakres}(Plik_a, \text{Katalog_A}, \text{Katalog_B}) \equiv \text{TRUE}^7$,
 - $\text{Zakres}(Plik_b, \text{Katalog_A}, \text{Katalog_B}) \equiv \text{FALSE}$
 - $\text{Stanowisko}(Podmiot_1, \text{Admin}) \equiv \text{FALSE}$,
 - $\text{Odgrywa}(Podmiot_1, \text{Pracownik}) \equiv \text{TRUE}$,

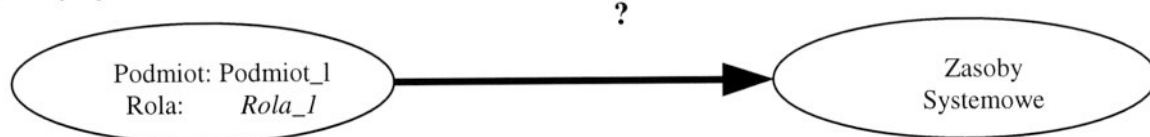
Sprawdzić:

Czy system kontroli dopuści do zrealizowania następujących żądań dostępu:

a) $Czytaj(Podmiot_1, Plik_a),$



b) $Czytaj(Podmiot_1, Plik_b).$



Rozwiązanie zadania:

Krok 1.

Po wykonaniu translacji semi-funkcyjnej na R1 i R2 otrzymujemy następujący zbiór klauzul dla programu logicznego:

- C1. $Zakres(x, Plik_a, Katalog_A, Katalog_B) \Leftarrow$
- C2. $\Leftarrow Zakres(x, Plik_b, Katalog_A, Katalog_B)$
- C3. $\Leftarrow Stanowisko(Podmiot_1, Admin)$
- C4. $Ogrywa(x, Podmiot_1, Pracownik) \Leftarrow$
- C5. $R(x, f(x)) \Leftarrow$
- C6. $Czytaj(f(x), pd, pl) \Leftarrow Zakres(x, pl, Katalog_A, Katalog_B),$
 $Odgrywa(x, pd, Pracownik)$
- C7. $\Leftarrow Stanowisko(x, pd, Admin), Przyznaj_role(f(x), pd, pl)$

Następnie kierujemy do programu pytania:

ad. a)

a) $\Leftarrow Czytaj(y, Podmiot_1, Plik_a)$

Postępując zgodnie z Algorytmem 6.1. otrzymujemy następującą sekwencję klauzul:

- 1. $\Leftarrow Zakres(x, Plik_a, Katalog_A, Katalog_B),$
 $Odgrywa(x, Podmiot_1, Pracownik)$ (C6 i a)
- 2. $\Leftarrow Odgrywa(x, Podmiot_1, Pracownik)$ (1. i C1)
- 3. $\Leftarrow$ (2. i C4)

⁷ wartości logiczne TRUE = prawda, FALSE=falsz

Algorytm zwraca odpowiedź TAK (otrzymano klauzulę pustą), tak więc akcja $Czytaj(Plik_a, Katalog_A)$ nie jest sprzeczna z przyjętymi uregulowaniami dla podmiotu $Podmiot_1$.

ad. b)

$b) \Leftarrow Czytaj(y, Podmiot_1, Plik_b)$

1.

$\Leftarrow Zakres(x, Plik_b, Katalog_A, Katalog_B),$
 $Odgrywa(x, Podmiot_1, Pracownik)$

(C6 i b)
2.

$\Leftarrow Zakres(x, Plik_b, Katalog_A, Katalog_B)$

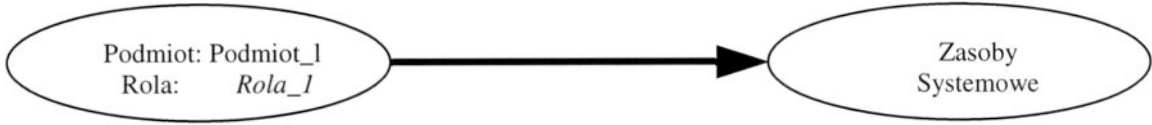
(1. i C4)
3.

STOP.

Algorytm zatrzymuje się w trzecim kroku, czyli akcja $Czytaj(Podmiot_1, Plik_a)$ jest sprzeczna z przyjętymi uregulowaniami dla podmiotu $Podmiot_1$.

Odpowiedź:

a) $Czytaj(Podmiot_1, Plik_a)$, - TAK (zaakceptowane żądanie dostępu)



b) $Czytaj(Podmiot_1, Plik_b)$, - NIE (odrzucone żądanie dostępu)



6.1. Podsumowanie.

W tym rozdziale został przedstawiony algorytm analizy spójności żądań podmiotów z przyjętą polityką oraz udowodnione zostało twierdzenie o dopuszczeniu do wykonania przez podmiot operacji dostępu. Algorytm ten wraz z twierdzeniem stanowią podstawę mechanizmu umożliwiającego weryfikację zgodności zgłaszanych przez podmioty żądań dostępu z obowiązującą w systemie polityką sterowania dostępem.

7. Mechanizmy zarządzania dostępem.

W tym rozdziale zdefiniowane zostaną mechanizmy zarządzania dostępem oparte na formalizmie *RMM*. W tej części procesu sterowania dostępem uwidaczniają się najbardziej zalety wykorzystania logiki matematycznej. Do najważniejszych elementów polityki RBAC z dziedziny zarządzania dostępem zalicza się proces definiowania i modyfikowania ról, hierarchii i ograniczeń związanych ze zdefiniowanymi rolami [Tho91], [SM94], [Kuh97], [FK92], [Loz91]. Dla większości współczesnych systemów istotna jest również możliwość decentralizacji procesu zarządzania. We wszystkich takich przypadkach, zachodzi konieczność ustalenia i sterowania podziałem kompetencji pomiędzy wyróżnionymi podmiotami odpowiedzialnymi za zarządzanie dostępem. Problemy oraz propozycja ich rozwiązania jest opisana w kolejnych punktach.

7.1. Spójność wewnętrzna ról.

Jednym z podstawowych założeń dla przedstawionego wcześniej *algorytmu 6.1.* badania spójności żądania dostępu z przyjętą polityką było to, aby opisujący politykę sterowania dostępem zbiór norm był wewnętrznie spójny. Spójność unormowań danej polityki powinna być weryfikowana w organizacji na kilku poziomach. Najwyższym poziomem jest poziom ogólnej polityki organizacji, na podstawie której powstają i którą powinny wspierać wszystkie inne polityki organizacji, takie jak na przykład: polityka kadrowa, polityka bezpieczeństwa, polityka sterowania dostępem, itp. [Car83] [Ino91], [CC98], [Cho97], [NO95].

Normy wyznaczające zbiór dopuszczalnych operacji dla poszczególnych podmiotów, muszą być spójne na dwóch następujących poziomach. Na poziomie pojedynczej roli, która stanowi najmniejszy zbiór norm jaki można przypisać każdemu z podmiotów oraz na poziomie roli w danym miejscu hierarchii ról obowiązującej w systemie. Te drugie zagadnienie zostanie opisane przy omawianiu problemów związanych z hierarchicznością ról.

Na potrzeby rozważań tego punktu założono, że każda rola jest autonomiczna, tzn. nie zależy i nie jest powiązana w jakikolwiek inny sposób od innych istniejących w systemie ról.

Można wyróżnić zadania, które są związane z procesem dodawania nowych norm do istniejącej roli, a mianowicie:

- a) zadanie rozszerzania zbioru akcji dozwolonych, zabronionych i obowiązkowych,
- b) zadanie rozwiązania problemu sprzeczności norm.

7.1.1. Zadanie rozszerzania zbioru akcji dozwolonych, zabronionych i obowiązkowych

Z zagadnieniem rozszerzania zezwolenia (zakazu) do wykonywania przez podmiot danej akcji dostępu, mamy odczynienia wtedy, kiedy dana rola w swoim zbiorze formuł zawiera już formułę opisującą zezwolenie na wykonanie danej akcji przy spełnieniu przez podmiot określonych warunków wstępnych.

Np. rola zawiera formułę zezwalającą podmiotom na tworzenie kopii zapasowych wszystkich plików systemowych pod warunkiem, że jest piątek.

$$\forall x \text{Dzień}(x) \wedge \text{Piątek}(x) \rightarrow \mathbf{P} \text{Twórz_kopię}(\text{Pliki_systemowe})$$

Chcemy zaś dodać podmiotom odgrywającym tę rolę możliwość tworzenia kopii zapasowych wszystkich plików systemowych również i każdego pierwszego dnia miesiąca.

$$\forall x \text{Nowy_miesiąc}(x) \rightarrow \mathbf{P} \text{Twórz_kopię}(\text{Pliki_systemowe})$$

Ogólnie możemy zapisać ten problem w następujący sposób:

Chcemy dodać do roli *R1* formułę $\text{War_2} \rightarrow \mathbf{P} \text{Akcja_1}$ podczas, gdy w roli tej zdefiniowana jest już formuła $\text{War_1} \rightarrow \mathbf{P} \text{Akcja_1}$.

W systemach z implikacją ścisłą, w przypadku kiedy zbiory wartości, dla których *War_1* i *War_2* są spełnione, byłyby rozłączne, otrzymalibyśmy sprzeczność przyjętych unormowań dla pewnego zakresu wartości zmiennych.

Np. w przypadku gdy $\text{War_1} \equiv \text{T}$, a $\text{War_2} \equiv \text{F}$, z pierwszej formuły wynikałoby, że $\mathbf{PAkcja_1} \equiv \text{T}$, podczas gdy druga z drugiej otrzymalibyśmy, że $\mathbf{PAkcja_1} \equiv \text{F}$, a więc sprzeczność.

W rozważanym przypadku, kiedy w *RMM* posługujemy się jedynie implikacją materialną, unikamy podobnego problemu. Jak wiadomo implikacja materialna przyjmuje wartość T, gdy jej poprzednik ma wartość logiczną F, dla dowolnej wartości następnika.

W związku z tym, w przypadku istnienia dwóch odrębnych warunków dotyczącej jednej akcji, otrzymujemy poszerzenie zbioru warunków dla których podmiot będzie mógł ją wykonywać. Takie podejście jest korzystne z punktu widzenia polityki RBAC z tego względu, iż ułatwia implementację mechanizmu dziedziczenia uprawnień z ról podrzędnych.

Wszystkie powyższe uwagi odnoszą się również do formuł określających akcje nakazane ($\mathbf{OAkcja_1}$), jak i zabronione ($\mathbf{FAkcja_1}$).

7.1.2. Zadanie rozwiązania problemu sprzeczności norm

W przypadku kiedy mamy do czynienia ze sprzecznymi normami, sytuacja może być trudniejsza do zinterpretowania w sposób jednoznaczny.

Źródłami potencjalnych sprzeczności w zdefiniowanej roli mogą być formuły postaci $\mathbf{PAkcja_1}$ i $\mathbf{FAkcja_1}$, które są spełnione dla takiego samego zbioru wartości występujących w nich zmiennych. W takiej sytuacji, nie byłoby możliwe podjęcie jednoznacznej decyzji o akceptacji, bądź też o odrzuceniu żądania wykonania tejże akcji, w oparciu o skojarzony z rolą zbiór formuł.

Analogiczną parą formuł, które mogą powodować powstanie w systemie uregulowań konfliktów są formuły: $\mathbf{OAkcja_1}$ i $\mathbf{FAkcja_1}$. Sposób wykrywania i rozstrzygania powstałych problemów w obydwóch przypadkach pozostaje taki sam.

Najpierw przeanalizujemy jakie są formalne i praktyczne konsekwencje z istnienia w jednej roli wzajemnie sprzecznych norm.

Założmy, że w definicji roli istnieją obok siebie dwie następujące formuły:

$$War_1 \rightarrow \mathbf{P}Akcja_1 \text{ i } War_2 \rightarrow \mathbf{F}Akcja_1.$$

Jeżeli domeny wartości zmiennych występujących w warunkach War_1 i War_2 dla których warunki te przyjmują wartość logiczną T, są rozłączne, to pomimo istnienia obok siebie potencjalnie sprzecznych uregulowań $\mathbf{P}Akcja_1$ i $\mathbf{F}Akcja_1$, w trakcie działania rzeczywistego systemu nie powinno jednak dochodzić konfliktów. Przykładem tego typu zależności mogą być formuły:

$$\begin{aligned} \forall x \text{ Dzień}(x) \wedge \text{Piątek}(x) &\rightarrow \mathbf{P} \text{ Twórz_kopię}(\text{Pliki_systemowe}) \\ \forall x \text{ Dzień}(x) \wedge \text{Poniedziałek}(x) &\rightarrow \mathbf{F} \text{ Twórz_kopię}(\text{Pliki_systemowe}) \end{aligned}$$

W każdym innym przypadku tzn. gdy istnieje taka wartość zmiennych, dla której jednocześnie War_1 i War_2 przyjmują wartość logiczną T, będziemy mieli możliwość zaistnienia sytuacji realnego konfliktu. W takim przypadku możliwe jest, że przy tych samych warunkach wstępnych algorytm 6.1. będzie zwracał różne rezultaty przy pytaniu o dopuszczalność wykonania akcji $Akcja_1$.

Na potrzeby definiowanego mechanizmu kontroli spójności założono, że:

- istnienie norm, dla których w całym zbiorze wartości zmiennych nie dochodzi do powstania sprzeczności - jest dopuszczalne,
- istnienie norm, dla których istnieje taki zbiór wartości zmiennych, dla którego dochodzi do konfliktu - nie jest dopuszczalne.

Poniższy algorytm oparty jest na automatycznej metodzie rezolucji, która pozwala znajdować formuły będące logiczną konsekwencją istniejącego zbioru formuł i nowowprowadzonego warunku, a które mogą być źródłem powstawania sprzeczności w zbiorze możliwych światów.

Algorytm 7.1. **BADANIA SPÓJNOŚCI NOWODODAWANEJ FORMUŁY Z ISTNIEJĄCYM ZBIOREM FORMUŁ.**

Niech formuła, której spójność z obowiązującym zbiorem formuł chcemy sprawdzić jest postaci: $Warunek \rightarrow Akcja$. W tym celu wykonujemy następujące czynności.

1. - Przygotowujemy wszystkie przyjęte jako zbiór norm formuły oraz nową formułę do potrzeb automatycznego wnioskowania, tzn. sprowadzamy je do postaci CNF , wykonujemy translację semi-funkcyjną, etc.
2. - Obowiązujące formuły zapisujemy jako treść programu logicznego w postaci klauzul Horna,
3. - Formułujemy zapytanie $\Leftarrow Warunek$,
 - a) **if** odpowiedź programu logicznego NIE
 - $Warunek$ jest sprzeczny z obowiązującymi ustaleniami, w związku z czym również formuła $Warunek \rightarrow Akcja$ jest niezgodna z przyjętym zbiorem norm,
 - Odrzucamy formułę $Warunek \rightarrow Akcja$ jako niespójną z obowiązującymi normami,
 - **return** NIE

STOP
 - b) **if** odpowiedź programu logicznego TAK
 - dodajemy do zbioru klauzul programu logicznego $Warunek \Leftarrow$,
4. - Uruchamiamy program pytaniem $\Leftarrow Akcja$,
 - a) **if** odpowiedź programu logicznego NIE,
 - Wniosek taki sam jak w punkcie 3.a)
 - b) **if** odpowiedź programu logicznego TAK,
 - Badana formuła nie jest sprzeczna z obowiązującymi normami,
 - Dodajemy formułę $Warunek \rightarrow Akcja$ do obowiązującego zbioru norm Reg .
 - **return** TAK

STOP

Twierdzenie 7.1. **O SPÓJNOŚCI NOWODODAWANEJ FORMUŁY Z ISTNIEJĄCYM ZBIOREM FORMUŁ.**

W wyniku operacji dodania nowej formuły do istniejącego zbioru formuł otrzymujemy zbiór spójny, wtedy gdy operację tę wykonamy zgodnie z algorytmem 7.1. i algorytm ten zwróci wartość TAK.

Dowód

Dowód poprawności rozumowania przedstawionego w algorytmie 7.1. oraz twierdzenia 7.1. oparty jest na dwóch następujących twierdzeniach rachunku predykatów [Bat94]:

- a) Jeżeli $B \in \text{Cn}(X \cup A)$ to $(A \rightarrow B) \in \text{Cn}(X)$,
- b) Jeżeli $A \in \text{Cn}(X)$ oraz $(A \rightarrow B) \in \text{Cn}(X)$, to $B \in \text{Cn}(X)$,

Podstawiamy $A \equiv \text{Warunek}$, $B \equiv \text{Akcja}$, $X \equiv \text{Reg}$. Przeglądając algorytm 7.1. stwierdzamy, że przebieg jego jest zgodny z podanymi warunkami w twierdzeniu z punktów a) i b).

q.e.d.

Przykład 7.1. ROZSZERZANIE ZBIORU FORMUŁ DEFINIUJĄCEGO ROLĘ.

Do zbioru formuł *Reg* dodajemy formułę RD:

Dane:

RD:

$$\forall pd \text{ Odgrywa}(pd, \text{Pracownik}) \rightarrow \mathbf{P}\text{Przysnaj_rolę}(pd, \text{Kierownik_A})$$

Reg:

$$\text{R1.} \quad \forall pd \text{ Stanowisko}(pd, \text{Admin}) \rightarrow \neg \mathbf{P}\text{Przysnaj_rolę}(pd, \text{Kierownik_A})$$

oraz

$$\text{Warunek} \equiv \forall pd \text{ Odgrywa}(pd, \text{Pracownik}),$$

$$\text{Akcja} \equiv \mathbf{P}\text{Przysnaj_rolę}(pd, \text{Kierownik_A}).$$

Pytanie:

Czy formuła RD jest spójna ze zbiorem formuł *Reg*?

Rozwiązanie:

W celu określenia odpowiedzi na powyższe pytanie stosujemy procedurę postępowania zgodną z algorytmem 7.1. w następujący sposób.

Krok 1.

$$\forall pd \text{ Odgrywa}(x, pd, \text{Pracownik}) \rightarrow \exists f \text{ Przysnaj_rolę}(f(x), pd, \text{Kierownik_A})$$

$$\forall pd \text{ Stanowisko}(x, pd, \text{Admin}) \rightarrow \neg \exists f \text{ Przysnaj_rolę}(f(x), pd, \text{Kierownik_A})$$

Krok 2.

Program:

$$\Leftarrow \text{Stanowisko}(x, pd, \text{Admin}), \text{Przysnaj_rolę}(f(x), pd, \text{Kierownik_A})$$

$$R(x, f(x)) \Leftarrow$$

$$\text{Odgrywa}(x, pd, \text{Pracownik}) \Leftarrow$$

Pytanie:

$\Leftarrow \text{Odgrywa}(x, pd, \text{Pracownik})$

Odpowiedź:

$\Leftarrow$

TAK,

Krok 3.

$\Leftarrow \text{Stanowisko}(x, pd, \text{Admin}), \text{Przypisz_rolę}(f(x), pd, \text{Kierownik_A})$

$R(x, f(x)) \Leftarrow$

$\text{Odgrywa}(x, pd, \text{Pracownik}) \Leftarrow$

Krok 4.

Pytanie:

$\Leftarrow \text{Przypisz_rolę}(f(x), pd, \text{Kierownik_A})$

Odpowiedź:

NIE

Odrzucamy formułę $\forall pd \text{ Odgrywa}(pd, \text{Pracownik}) \rightarrow \mathbf{P} \text{Przypisz_rolę}(pd, \text{Kierownik_A})$ jako niezgodną z przyjętym zbiorem reguł *Reg*.

STOP

Odpowiedź:

Formuła RD nie jest spójna ze zbiorem formuł *Reg* i nie należy dopuścić do jej dodania do zbioru *Reg*.

7.2. Hierarchia ról.

Jak wykazano we wcześniejszych rozważaniach, możliwość budowania hierarchicznych struktur sterowania dostępem, które są w stanie modelować hierarchię obowiązującą w danej organizacji, jest jedną z poważniejszych zalet polityki RBAC. Zgodnie z założeniem, hierarchia ról powinna umożliwiać dziedziczenie właściwości związanych z rolą podrzędną przez rolę nadrzędną [BK85], [San98], [GGF98], [San90]. W literaturze opisującej różnorodne implementacje polityki RBAC zazwyczaj nie precyzuje się w jaki sposób ta właściwość ma być realizowana oraz jakie są związane z nią dodatkowe wymagania, warunki wstępne, itp.

Jako jedno z podstawowych wymagań związanych z tworzeniem hierarchii ról można wyróżnić żądanie zapewnienia spójności zbioru norm powstałego na drodze dziedziczenia uprawnień przez role nadrzędne. Większość autorów nie wspomina o tym zagadnieniu, a przez to wydaje się zakładać, że nie istnieją konflikty pomiędzy dziedziczonymi i posiadanymi uprawnieniami [Bar95b], [FCK95], [FK92], [SCFY94]. Takie założenie jest do

przyjęcia jedynie wtedy, gdy przyjmiemy, iż na drodze uważnego projektowania ról i analizowania powiązań pomiędzy nimi, wyeliminowano wszelkie możliwe konflikty. Takie założenie, zwłaszcza w stosunku do systemów rozległych, w których dynamicznie ulegają zmianie definicje ról, jest w praktyce nie do przyjęcia. Dlatego też, potrzebujemy odpowiednich mechanizmów, które to umożliwiałyby analizę konsekwencji wprowadzanych do hierarchii ról zmian.

Poniżej przedstawiono mechanizm budowania i sterowania hierarchią ról oparty na formalizmie *RMM*.

Zdefiniujmy w języku *RMM* dodatkowy predykat o postaci np. *Dziedziczy(x)*, gdzie x jest zmienną, która przyjmuje jako wartość nazwy zdefiniowanych w systemie ról. Predykat ten zwraca wartość logiczną T w przypadku, gdy aktywna rola jest rodzicem roli x .

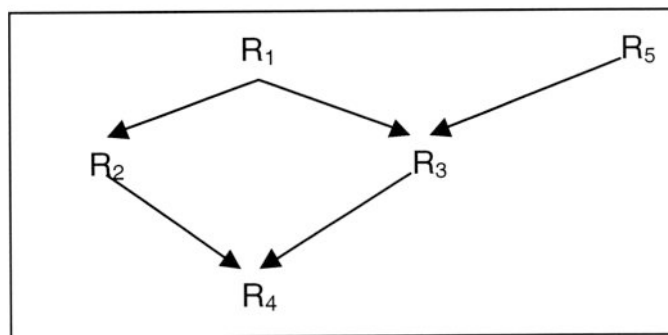
Na przykład dla przedstawionego poniżej schematu, niech role R_1, R_2, R_3, R_4 , obok formuł definiujących uprawnienia przysługujące podmiotom odgrywającym te role, zawierają odpowiednio również i formuły:

R_1 : *Dziedziczy(R₂)*, *Dziedziczy(R₃)*;

R_2 : *Dziedziczy(R₄)*;

R_3 : *Dziedziczy(R₄)*;

Na potrzeby programów logicznych formuły te tłumaczone będą na postać klauzulową postaci: *Dziedziczy(x) ⇐*.



rys 7.1. Przykładowa hierarchia ról.

Po dołączeniu do roli formuły *Dziedziczy(x)*, w najprostszym przypadku, rola ta powinna od tego momentu łączyć w sobie wszystkie uprawnienia związane z nią samą i z rolą x oraz z ewentualnymi potomkami roli x . Dla sytuacji opisanej powyżej w momencie dołączenia do roli R_5 formuły *Dziedziczy(R₁)*, zbiór jej możliwości rozszerza się o możliwości dostępne dla podmiotów z poziomu roli R_3 i R_4 . Dodatkowym założeniem będzie więc, że nie ograniczamy liczby poziomów hierarchii, przez które możliwe jest dziedziczenie uprawnień.

W wielu przypadkach korzystne jest jednak zapewnienie tego, aby nie wszystkie uprawnienia związane z daną rolą mogły być każdorazowo dziedziczone przez dowolną

nadrzędną względem niej rolę. Dlatego też, niekiedy zachodzi konieczność ograniczenia możliwości przepływu wraz z mechanizmem dziedziczenia określonego zbioru uprawnień.

W zdefiniowanym modelu polityki sterowania dostępem, możliwe jest zastosowanie następującego zabiegu ograniczającego przepływ uprawnień. Do warunku wykonania danej operacji (poprzednika implikacji) dodajemy predykat:

$$Odgrywa(Podmiot, Rola_bieżąca),$$

gdzie:

Podmiot - jest nazwą podmiotu zgłaszającego żądanie wykonania danej operacji,
Rola_bieżąca - jest stałą, określającą nazwę roli w której zdefiniowany jest warunek.

Jeżeli w roli R_3 zdefiniowana jest formuła:

$$F1: War_1 \rightarrow \mathbf{P} Akcja_1;$$

oraz nie chcemy aby możliwość wykonania operacji *Akcja_1* była dostępna dla podmiotów, które odgrywają role dziedziczące R_3 modyfikujemy $F1$ w sposób opisany powyżej i otrzymujemy formułę:

$$F1': \forall P Odgrywa(P, R_3) \wedge War_1 \rightarrow \mathbf{P} Akcja_1.$$

Zauważmy, że w analogiczny sposób możemy formułować warunki, które pozwalają na o wiele bardziej elastyczne zarządzanie prawami w hierarchii niż tylko w oparciu o proste dziedziczenie. Równie dobrze możemy zdefiniować predykat:

$$Zawiera_się(Podmiot, R_top, R_down),$$

który określać będzie przedział w hierarchii dla uprawnionego dziedziczenia danej formuły.

Należy jednak pamiętać o tym, że wprowadzenie zbyt wielu różnorodnych warunków, które powodują omijanie ustalonej hierarchii dziedziczenia uprawnień, powodować będzie utratę przejrzystości unormowań, a przez to zwiększać będzie prawdopodobieństwo wystąpienia błędu. Z kolei, nadmierne ograniczanie dziedziczenia, czy też zezwalanie na dziedziczenie uprawnień rolom niepowiązanym hierarchicznie z rolą bieżącą, kwestionuje istnienie hierarchii jako elementu wspierającego proces sterowania dostępem. Niemniej jednak potencjalnie szeroki zakres możliwych do wprowadzenia modyfikacji względem istniejącej hierarchii, świadczy o elastyczności zarówno polityki RBAC, jaki i proponowanego przeze mnie w tej pracy modelu.

Twierdzenie 7.2. O OGRANICZENIU DZIEDZICZENIA UPRAWNIENÍ.

Zamiana warunku War_I w formule:

$$War_I \rightarrow \mathbf{P} Akcja_I,$$

na koniunkcję warunków:

$$War_I \wedge Odgrywa(Podmiot, Rola_bieżąca),$$

gdzie:

- | | |
|-----------------|--|
| $Podmiot$ | - nazwa podmiotu zgłaszającego żądanie wykonania danej operacji, |
| $Rola_bieżąca$ | - stała określającą nazwę roli w której zdefiniowany jest warunek, |
| $Odgrywa(x,y)$ | - predykat przyjmujący wartość logiczną TRUE wtedy i tylko wtedy, gdy podmiot x odgrywa rolę y . |

powoduje, że uprawniony do wykonywania operacji $Akcja_I$ jest jedynie podmiot odgrywający rolę $Rola_bieżąca$.

Dowód.

Niech:

- RP - rola, w której zamieniono formułę: $War_I \rightarrow \mathbf{P} Akcja_I$, na formułę:
 $\forall P Odgrywa(P, RP) \wedge War_I \rightarrow \mathbf{P} Akcja_I$,
- RN - rola nadrzędna względem RP (RN dziedziczy RP),

W roli RN i w żadnej podrzędnej roli względem RN i RP nie występuje warunek:

$$War_2 \rightarrow \mathbf{P} Akcja_I$$

Prześledźmy działanie algorytmu badania zgodności żądania wykonania danej operacji, z ustalonym zbiorem norm określających możliwości podmiotu.

- w klauzulach programu logicznego formuła $\mathbf{P} Akcja_I$ występuje po lewej stronie tylko raz, w klauzuli: $\mathbf{P} Akcja_I \Leftarrow War_I, Odgrywa(P, RP)$,
- wartość logiczna $Odgrywa(P, RP) \equiv F$, czyli w programie logicznym występuje klauzula: $\Leftarrow Odgrywa(P, RP)$,
- nie istnieje możliwość zredukowania literału $Odgrywa(P, RP)$ w klauzuli: $\mathbf{P} Akcja_I \Leftarrow War_I, Odgrywa(P, RP)$,

Z ostatniego faktu wnioskujemy, że program logiczny na pytanie $\Leftarrow Akcja_I$ da odpowiedź NIE, co oznacza, że podmiot nie może wykonać operacji $Akcja_I$.

q.e.d.

Proces dołączania przez nadzorcę formuły $Dziedziczy(x)$ do zbioru formuł opisujących daną rolę powinien aktywować procedurę badania spójności. Poniżej przedstawię algorytm określający prawidłową operację tworzenia hierarchii ról, która zachowuje spójność zdefiniowanych unormowań.

Algorytm 7.2. BUDOWANIA HIERARCHII RÓL.

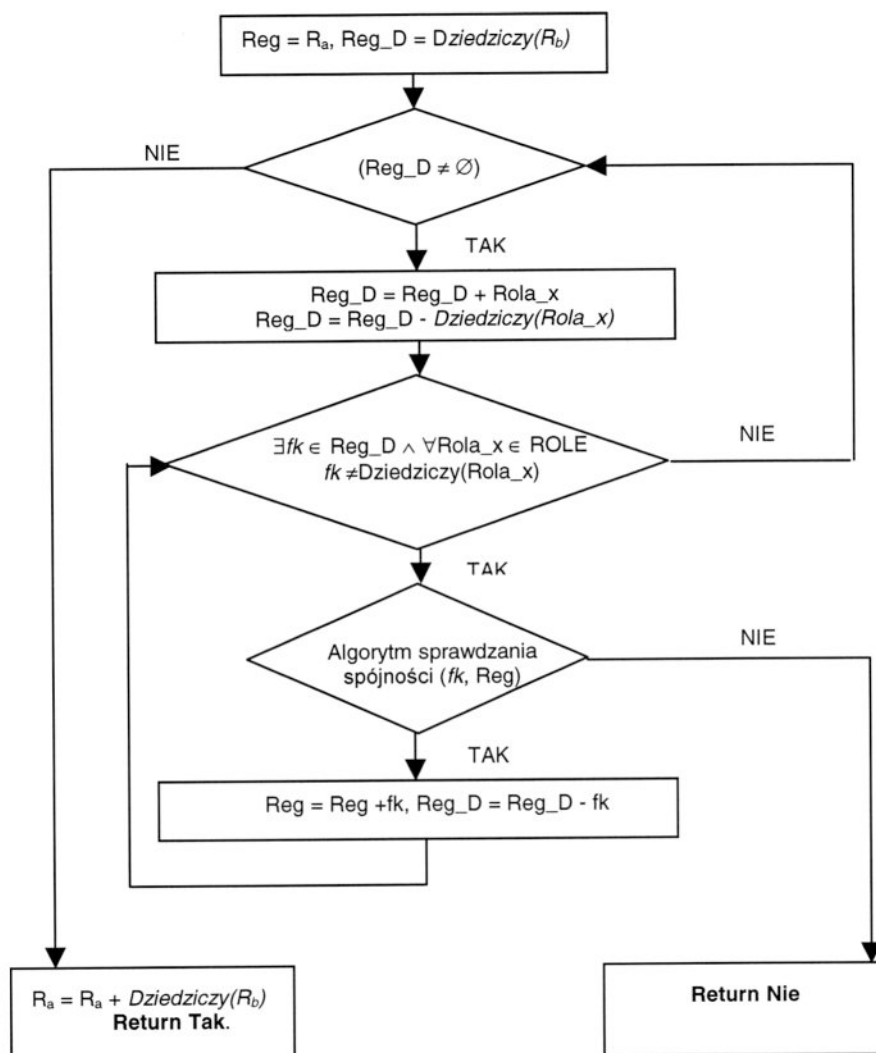
Zadanie:

Do hierarchii ról dodajemy rolę R_a , tak aby była ona rolą bezpośrednio nadrzędną wobec roli R_b .

Dane:

ROLE	- zbiór nazw wszystkich do tej pory utworzonych ról
Reg	- tymczasowy zbiór przechowujący formuły roli nadrzędnej
Reg_D	- tymczasowy zbiór przechowujący formuły ról podrzędnych
Rola_x	- zmienna przechowująca nazwę aktualnie sprawdzanej roli z istniejącej do tej pory hierarchii
fk	- zmienna przechowująca aktualnie badaną formułę,
$Dziedziczy(Rola)$	- predykat określający zależności pomiędzy rolami w hierarchii. - <i>Jeżeli rola R_a posiada w zbiorze swoich formuł predykat $Dziedziczy(R_b)$ to znaczy, że jest ona bezpośrednio nadrzędną rolą w hierarchii względem roli R_b</i>

1. $Reg = R_a, Reg_D = Dziedziczy(R_b)$
2. **While** ($Reg_D \neq \emptyset$) **Do**
3. $\forall Rola_x \in ROLE$
If $Dziedziczy(Rola_x) \in Reg_D$
Then
 $Reg_D = Reg_D + Rola_x$
 $Reg_D = Reg_D - Dziedziczy(Rola_x)$
4. **While** ($\exists fk \in Reg_D \wedge \forall Rola_x \in ROLE \, fk \neq Dziedziczy(Rola_x)$) **Do**
5. **If** Algorytm sprawdzania spójności (fk, Reg) == TAK
Then
 $Reg = Reg + fk$
 $Reg_D = Reg_D - fk$
Else
 Komunikat: „Wykryto sprzeczności w budowanej hierarchii!”
Return NIE
6. $Ra = Ra + Dziedziczy(Rb)$
7. Komunikat: „Budowa hierarchii zakończona pomyślnie!”
8. **Return Tak.**



rys. 7.2. Schemat blokowy algorytmu 7.2.

Twierdzenie 7.3. O SPÓJNOŚCI HIERARCHII RÓL.

Algorytm budowania hierarchii ról zapewnia spójność konstruowanych w oparciu o ten algorytm hierarchii ról.

Dowód.

Algorytm 7.2. polega na sekwencyjnym dodawaniu do spójnego zbioru formuł Reg , skojarzonego z dodawaną do hierarchii rolą, formuł pochodzących z ról podrzędnych. Spójność każdej nowododanej formuły ze zbiorem formuł roli jest analizowana zgodnie z Algorytmem 7.1, którego skuteczność wykazano wcześniej. W związku z tym, można stwierdzić, iż Twierdzenie 7.3. zachodzi.

q.e.d.

Algorytm badający spójność hierarchii ról jest efektywny w tym sensie, że jest w stanie wykryć wszelkie pojawiające się pomiędzy unormowaniami konflikty. Niemniej jednak charakteryzuje się on znaczną złożonością obliczeniową, która powodować może duże wymagania czasowe związane z jego działaniem. Ponieważ jednak konieczność weryfikowania spójności hierarchii ról nie zachodzi w rzeczywistych warunkach często oraz może być wykonywana na dedykowanym do tego zadania serwerze, a zmiany wprowadzane mogą być dopiero po sprawdzeniu czy zostanie zachowana spójność systemu.

Kolejna ważna kwestia związana z hierarchicznością ról oraz badaniem spójności ról w hierarchii jest to, że po powiązaniu ról w hierarchię, wszelkie zmiany dotyczące pojedynczej roli przestają mieć wyłącznie lokalny charakter. Oznacza to, że przy modyfikowaniu zbioru formuł związanych z rolą włączoną w hierarchię ról, nie wystarczające będzie sprawdzenie jedynie tego czy nie została naruszona spójność unormowań tej roli. Dla zapewnienia poprawności działania systemu, konieczna jest weryfikacja wszystkich ról powiązanych ze zmienianą rolą. Wyjątek mogą stanowić tylko takie zmiany, które mają być z założenia lokalne, takie jak np. dodanie formuły nie podlegającej dziedziczeniu i która nie może być spełniona dla podmiotów nie będących bezpośrednio powiązanymi z daną rolą (np. wcześniej rozpatrywany przypadek formuły $\forall P \text{ Odgrywa}(P, R_3) \wedge \text{War}_I \rightarrow \mathbf{P} \text{ Akcja}_I$).

7.3. Ograniczenia.

Rodzaje ograniczeń i ich znaczenie dla polityki RBAC opisane pokrótce zostały w rozdziale 3. i 4., a szersze omówienie tych zagadnień można znaleźć w literaturze [San88], [San96], [SCFY94]. Poniżej przedstawię sposób definiowania i sterowania ograniczeniami w oparciu o wcześniej zdefiniowane mechanizmy oraz własności *RMM*.

Wyróżnia się trzy podstawowe kategorie ograniczeń:

- wzajemne wykluczanie się,
- wzajemne wymaganie się,
- ograniczenia na ilość aktywnych odwołań.

Definiowanie ograniczeń na bazie prezentowanych w tej pracy mechanizmów polega na dodaniu do roli określonych formuł warunkujących zgodność modyfikowanych ról z warunkiem nałożonym przez dane ograniczenie.

7.3.1. Ograniczenia na ilość aktywnych odwołań.

Możliwość skutecznego kontrolowania maksymalnej liczby podmiotów aktywnych w danej roli wymaga istnienia w systemie globalnego licznika, którego zadaniem byłoby zliczanie ilości przypisań podmiotów do każdej z ról. W takim wypadku możliwe jest stworzenie listy formuł określających maksimum jakie może osiągnąć liczba aktywnych w danej roli podmiotów. Wtedy, stanowiłaby ona listę warunków wstępnych, które są analizowane każdorazowo przed przypisaniem podmiotu do roli. Lista ta mogłaby być następującej postaci:

Card_list: Card_R1(15), Card_R2(5), Card_R3(1), Card_R4(7).

gdzie:

Card_Rn(x) - predykat, zwracający wartość logiczną TAK w przypadku kiedy liczba podmiotów aktywnych w roli *Rn* jest mniejsza od *x* oraz wartość NIE każdym innym przypadku.

Przy założeniu, że operacja sprawdzania warunku osiągnięcia maksymalnej liczby aktywnych podmiotów w roli oraz ewentualnego przyznania roli podmiotowi jest operacją niepodzielną, możliwe jest traktowanie warunków: *Card_R1(15), ...*, jak klauzul, których zgodność można sprawdzać przy użyciu prostej rezolucji np. w sposób następujący:

Card_list: Card_R1(15) ⇐, Card_R2(5) ⇐, Card_R3(1) ⇐, Card_R4(7) ⇐.

Pytanie: $\Leftarrow Card_R1(Act)$,

W takim przypadku, gdy program logiczny da odpowiedź TAK, oznacza to, że osiągnięto maksimum i nie można aktywować nowego podmiotu w danej roli. W przypadku uzyskania odpowiedzi NIE, możemy dokonać zamierzonego przypisania. Algorytm ten jest skuteczny tylko wtedy, gdy od chwili zadania jednego pytania do zbioru *Card_list*, blokowany jest dostęp do niego. Zwalniany powinien być on dopiero w chwili uzyskania odpowiedzi TAK, bądź po przypisaniu nowego podmiotu do roli. Jeżeli nie zostałby zachowany powyższy warunek, można by spowodować przekroczenie maksymalnej dozwolonej liczby powiązań i w efekcie utracić kontrolę nad liczbą podmiotów aktywnych w danej roli.

Zaletą przedstawionego podejścia do ograniczeń na liczbę aktywnych podmiotów w danej roli jest zgodność z filozofią działania polityki RBAC oraz wykorzystanie tych samych narzędzi analizy, tzn. logiki formalnej. Niemniej jednak w tym przypadku bardziej efektywne pod względem zapotrzebowania na czas wykonania operacji sprawdzenia warunków, byłoby proste zapisanie maksymalnych ilości aktywnych podmiotów dla każdej roli w kilku rejestrach, których zawartość byłaby aktualizowana każdorazowo przy aktywacji i dezaktywacji podmiotów i których wartość można by sprawdzić w dowolnej chwili.

7.3.2. Role wzajemnie wykluczające się.

Realizacja ograniczenia dla ról wzajemnie wykluczających się, w przeciwieństwie do wcześniej omawianego ograniczenia na ilość aktywnych powiązań, nie wymaga wykorzystywania żadnych zewnętrznych w stosunku do danej roli mechanizmów. Warunki ograniczające przypisanie danej roli podmiotowi, któremu już wcześniej została przyznana inna rola, można zamieścić bezpośrednio w zbiorze formuł określających normy związane z rolą.

Można w tym celu wykorzystać stosowany już wcześniej predykat $Odgrywa(Podmiot, Rola)$, który zwraca wartość logiczną T, dla podmiotów z aktywną rolą $Rola$.

Niech:

R_Nowa - definiowana rola,

$Role_WW$ - zbiór ról będących w relacji wzajemnego wykluczania się z rolą R_Nowa

Wystarczy zatem dołączyć do zbioru formuł definiujących rolę R_Nowa formułę $\mathbf{F}Odgrywa(Podmiot, Rola_I)$, gdzie $Rola_I \in Role_WW$, aby móc odrzucać wszystkie żądania przyznania tej roli podmiotom ogrywającym już rolę $Rola_I$.

Aby jednak dopuścić do powiązania z tą rolą wszystkie inne podmioty, które nie są powiązane z żadną z ról ze zbioru $Role_WW$, należy dołączyć do definicji roli R_Nowa formułę $\mathbf{P}Odgrywa(Podmiot, R_Zmienna)$, gdzie $R_Zmienna$ jest zmienną podlegającą procesowi unifikacji w trakcie stosowania metody rezolucji.

Powyżej zdefiniowane ograniczenie ma sens jedynie wtedy, gdy każdorazowo przed przyznaniem danemu podmiotowi nowej roli jest sprawdzenie czy rola ta nie posiada ograniczenia na jej przyznawanie. W tym celu, przed przyznaniem podmiotowi nowej roli, np. R_Nowa , należy wygenerować w jego imieniu zapytanie w postaci klauzuli $\Leftarrow Odgrywa(Podmiot, R_bieżąca)$. Jeżeli otrzymamy odpowiedź TAK to oznacza, że podmiot może odgrywać jednocześnie rolę $R_bieżąca$ i R_Nowa . Jeżeli otrzymamy odpowiedź NIE, oznacza to, że $R_bieżąca$ i R_Nowa są w relacji wzajemnego wykluczania się i należy przerwać proces przyznawania nowej roli podmiotowi oraz wygenerować odpowiedni komunikat o błędzie.

Twierdzenie 7.4. O WZAJEMNYM WYKLUCZANIU SIĘ.

Jeżeli dowolny zbiór formuł Reg roli, dla dowolnej wartości $Rola_I$ zawiera formuły:

F1: $\mathbf{F}Odgrywa(Podmiot, Rola_I)$,

F2: $\mathbf{P}Odgrywa(Podmiot, R_Zmienna)$,

to zachodzi:

$$\forall P \text{ } Odgrywa(P, Rola_I) \notin Cn(Reg),$$

$$\forall Rola_Zmienna \neq Rola_I \text{ } Odgrywa(P, Rola_Zmienna) \in Cn(Reg).$$

Dowód.

Analizując zachowanie się programu logicznego po translacji semi-funkcyjnej formuł $\mathbf{FOdgrywa(Podmiot,Rola_1)}$ i $\mathbf{POdgrywa(Podmiot,R_Zmienna)}$, doprowadzeniu ich do postaci klauzulowej oraz zastosowaniu metody rezolucji dla pytania $\Leftarrow \mathbf{Odgrywa(Podmiot,Rola_Zmienna)}$ i pytania $\Leftarrow \mathbf{Odgrywa(Podmiot,Rola_Nowa)}$, otrzymujemy, iż jego odpowiedź jest zgodna z treścią Twierdzenia 7.4., co w kontekście niezawodności metody rezolucji kończy dowód.

q.e.d.

Zgodnie z Twierdzeniem 7.3. dodanie do zbioru formuł $\mathbf{FOdgrywa(Podmiot,Rola_1)}$ i $\mathbf{POdgrywa(Podmiot,R_Zmienna)}$ jest warunkiem wystarczającym, aby móc kontrolować relację wzajemnego wykluczania się ról w opisywanym systemie sterowania dostępem.

7.3.3. Role wymagające siebie nawzajem.

Zasada działania ograniczenia na role wzajemnie wymagające się jest podobna, z dokładnością co do warunku do tej, która opisywała role wzajemnie wykluczające się. W tym przypadku rola nie może zostać przyznana podmiotowi, jeżeli ten nie został wcześniej powiązany z inną, ściśle określoną rolą. Model formalny tego warunku jest konstruowany analogicznie jak w poprzednim przypadku z tym, że tym razem dodawana jest formuła postaci $\mathbf{OOdgrywa(Podmiot,Rola_1)}$, gdzie $Rola_1$ jest nazwą wymaganej roli. W odróżnieniu jednak od warunku wzajemnego wykluczania się ról nie możemy jednak dodawać dodatkowych warunków typu $\mathbf{POdgrywa(Podmiot,R_Zmienna)}$. Algorytm przyznawania ról i sprawdzania warunków wstępnych (ograniczeń) jest również taki sam jak poprzednio.

Twierdzenie 7.5. O WZAJEMNYM WYMAGANIU SIĘ RÓL.

Jeżeli dowolny zbiór formuł roli zawiera formułę:

$$\mathbf{OOdgrywa(Podmiot,Rola_1)},$$

to formuła

$$\mathbf{Odgrywa(P,Rola)} \in \mathbf{Cn(Reg)}$$

wtedy i tylko wtedy, gdy dla podmiotu P rola $Rola_1$ należy do zbioru jego aktywnych ról.

Dowód.

Po translacji semi-funkcyjnej formuły $\mathbf{OOdgrywa(Podmiot,Rola_1)}$ i zapisaniu jej w postaci klauzulowej, dla pytania $\Leftarrow \mathbf{Odgrywa(P,Rola)}$ program logiczny zwraca wartość logiczną TAK wtedy i tylko wtedy, gdy dla podmiotu P istnieje $Rola = Rola_1$, a stąd wynika, że $\mathbf{Odgrywa(P,Rola)} \in \mathbf{Cn(Reg)}$.

q.e.d.

7.3.4. Role z jednoczesnym ograniczeniami na wzajemne wykluczanie się i wymaganie innej roli.

W celu skonstruowania ograniczenia z jednocześnie spełniającego warunki wzajemnego wykluczania się z pewnym zbiorem ról i wymaganie odgrywania innego zbioru ról, wystarczy do roli dodać listę formuł wyznaczających te warunki, a sformułowanych tak jak to opisano w poprzednich punktach.

To znaczy, że do definicji roli *Nowa_rola* dodajemy formuły:

$$\mathbf{F}Odgrywa(Podmiot, R_1), \mathbf{F}Odgrywa(Podmiot, R_2), \mathbf{F}Odgrywa(Podmiot, R_3), \dots$$

dla wszystkich ról $R_1, R_2, R_3, \dots$ będących w relacji wzajemnego wykluczania się z rolą *Nowa_rola* oraz formuły:

$$\mathbf{O}Odgrywa(Podmiot, O_1), \mathbf{O}Odgrywa(Podmiot, O_2), \mathbf{O}Odgrywa(Podmiot, O_3), \dots$$

dla wszystkich ról $O_1, O_2, O_3, \dots$ wymaganych jako warunek wstępny dla podmiotu aby móc aktywować się również i w roli *Nowa_rola*.

W takim przypadku warunkiem spójności tych uregulowań jest, aby spełniony był warunek:

$$\forall R_n, O_m \Rightarrow R_n \neq O_m.$$

Jedyną różnicą pomiędzy takim złożonym ograniczeniem, a opisywanymi w poprzednich punktach ograniczeniami, jest brak warunków $\mathbf{P}Odgrywa(Podmiot, R_1), \dots$, charakterystycznych dla definicji ról wzajemnie się wykluczających. Warunek ten bowiem, pozostawałby w sprzeczności z formułami $\mathbf{O}Odgrywa(Podmiot, O_1), \dots$

7.4. Mechanizmy definiowania powiązań rola-prawa dostępu i rola-podmiot.

Ta kategoria mechanizmów zarządzania dostępem, stanowi o tym kto i za co jest odpowiedzialny oraz jakimi środkami dysponuje, aby realizować stojące przed nim zadania. Do tych mechanizmów zaliczamy:

- definiowanie i usuwanie podmiotów i obiektów w systemie,
- definiowanie i usuwanie ról (powiązanie rola-prawa dostępu),
- asocjacja ról z podmiotami (powiązanie rola-podmiot).

7.4.1. Definiowanie i usuwanie podmiotów i obiektów.

Na poziomie opisu systemu w języku formalnym, każda operacja utworzenia podmiotu/obiektu powinna być powiązana z utworzeniem pewnej stałej nazwowej, która będzie reprezentowała ten podmiot/obiekt na poziomie języka. Może towarzyszyć temu procesowi również proces definiowania podstawowych własności nowotworzonego podmiotu/obiektu, tak aby, można było poprawnie zidentyfikować i zinterpretować jego typ.

Np. gdy tworzymy nowy plik, to w sposób naturalny zostaje mu przypisana pewna nazwa taka jak *Dane.dat*, *Program.zip*, itp. Aby system mógł jednak rozpoznać, że pod stałą *Dane.dat* kryje się plik, może być konieczne dodanie formuły $Plik(Dane.dat) \equiv T$, a jeśli jest to plik tylko do odczytu to w zamian tego, formuły postaci:

$$Tylko_odczyt(Dane.dat) \wedge Plik(Dane.dat) \equiv T$$

w zależności od potrzeb i mocy ekspresji języka.

Jeżeli istnieją specjalne funkcje tworzące i usuwające podmioty i obiekty, to możemy formułować w języku *RMM* warunki pod jakimi mogą administratorzy systemu korzystać z tych funkcji. Jest to proces analogiczny do definiowania ról dla podmiotów nie będących administratorami.

Dla samego procesu sterowania dostępem operacja tworzenia podmiotu nie jest krytyczna, wtedy gdy nowoutworzony podmiot nie dysponuje żadnymi uprawnieniami. Dopiero proces przyznawania uprawnień powinien być uwzględniony przez proponowany model, tak aby umożliwiać kontrolowanie integralności całego systemu.

7.4.2. Definiowanie i usuwanie ról.

Zdefiniowanie nowej roli wymaga określenia jej charakterystyki (tzn. związanych z nią możliwości, ograniczeń, itp.) oraz określenia jej miejsca w hierarchii. Proces ten powinien zaczynać się od konstrukcji roli i weryfikacji jej spójności wewnętrznej. Następnie można przystąpić do rekonstrukcji hierarchii z uwzględnieniem nowoutworzonej roli. Wstawiając nową rolę do struktury określającej hierarchię powiązań pomiędzy rolami, należy wskazać rolę bezpośrednio podrzędną względem tworzonej oraz ewentualnie zmienić ustawienia roli podrzędnej w roli nadrzędnej względem dodawanej roli. Po aktualizacji powiązań konieczne jest dokonanie weryfikacji spójności hierarchii ról (mechanizmy wspierające ten proces zostały opisane wcześniej).

Konieczne jest również rozgraniczenie kompetencji podmiotów administracyjnych co do możliwości tworzenia i usuwania ról z systemu sterowania dostępem. Możliwe są rozwiązania bazujące na podziale ról na domeny lub poziomy, których administrowanie może być zlecone określonym podmiotom.

Proponowany mechanizm wspierający proces tworzenia i usuwania ról opiera się na podziale zbioru wszystkich ról na poziomy oraz na domeny. Poziom, jest związany z każdą z ról powinien być odpowiedzialnych za odzwierciedlenie hierarchicznej specyfiki powiązań pomiędzy rolami, tzn. faktu, że można wyróżnić role nadrzędne, posiadające zazwyczaj większy zbiór uprawnień niż role względem nich podrzędne. Jeżeli w systemie mamy kilka podmiotów odpowiedzialnych za zarządzanie dostępem, konieczne może okazać się rozróżnienie ich kompetencji co do możliwości ingerencji w strukturę uprawnień tzn. hierarchię ról.

Praktyczny sposób implementacji takiego ujęcia zarządzania rolami, wykorzystujący język modelu *RMM* oraz opisane wcześniej mechanizmy automatyzacji wnioskowania mogą wyglądać w następujący sposób.

Definiujemy dodatkowy predykat:

$$Poziom(R, lev),$$

gdzie:

- R - jest zmienną określającą nazwę roli,
 lev - określa poziom roli.

$Poziom(R, lev)$ zwraca wartość T wtedy i tylko wtedy, gdy poziom roli $R=lev$. Poziomy ról powinny być zdefiniowane jako zmienne globalne oraz zakładamy, że rola może dziedziczyć po roli o poziomie nie większym niż jej własny. Na tej podstawie, możemy zdefiniować następujący zbiór formuł określających możliwości podmiotu administracyjnego.

RADM_1:

F1: $Poziom(R, 15) \rightarrow \mathbf{P}Usun(R);$

F2: $\mathbf{P}Utworz(R, 15);$

F3: $Poziom(R1, 15) \wedge Poziom_dziecka(R2, lev) \rightarrow \mathbf{P}Dodaj_dziedzicz(R1, R2).$

Gdzie predykaty $Utworz(R, lev)$, $Dodaj_dziedzicz(R1, R2)$ oznaczają dodatkowe akcje związane z usuwaniem ról i dodawaniem własności dziedziczenia do istniejącej roli. Predykat $Poziom_dziecka(R, lev)$ sprawdza czy spełniony jest warunek ograniczający możliwości dziedziczenia pomiędzy rolami w hierarchii.

Należy pamiętać o tym, że akcje typu $Usun()$, $Dodaj()$ muszą być poprzedzone wykonaniem algorytmu weryfikacji spójności hierarchii ról opisanym przez Algorytm 7.2. Oprócz poziomów, można również powiązać każdą rolę z określoną grupą co może pozwolić na jeszcze bardziej precyzyjne rozgraniczenie kompetencji podmiotów administracyjnych. W grupy możemy łączyć role związane np. poprzez odpowiedzialność za wykonanie określonego zadania w organizacji czy też przez fizyczną lokalizację podmiotów odgrywających te role. Formalizacja pojęcia grupy na bazie języka *RMM* przebiega niemalże identycznie jak w przypadku poziomów ról. Na przykład możemy zdefiniować predykat $Grupa(R, gr)$ badający przynależność roli R do grupy gr . W takim wypadku możemy łączyć w warunkach określających zakres kompetencji roli administracyjnej, żądanie przynależności do grupy i do określonego poziomu np.:

$$Grupa(R, Technicy) \wedge Poziom(R, 7) \rightarrow \mathbf{P}Usun(R).$$

Usunięcie roli może w konsekwencji przynieść uszczuplenie możliwości ról na wyższym poziomie. Nie da się jednak tego efektu wyeliminować zupełnie. Konieczne jest natomiast sprawdzenie, czy usunięcie danej roli nie spowoduje błędów w istniejących ograniczeniach. Należy zatem odnaleźć wszystkie ograniczenia związane z daną rolą.

Proces usuwania ról niesie ze sobą konsekwencje nie tylko dla hierarchii ról i ról przez hierarchię związanych z usuwaną rolą, ale i również ma on wpływ na role powiązane z tą rolą na drodze ograniczeń. Wcześniej opisany mechanizm implementacji ograniczeń w języku *RMM* wyznacza również sposób weryfikowania spójności ograniczeń po usunięciu jednej roli z systemu. Mianowicie, ograniczenie *wzajemnego wykluczania się* jak i *wymagania się nawzajem* ról było tworzone poprzez dodanie do definicji ról któreś z formuł:

$$\mathbf{POdgrywa}(P,Rola), \mathbf{O}Odgrywa(P,Rola), \mathbf{FOdgrywa}(P,Rola).$$

Aby zatem zidentyfikować, które z istniejących ról były powiązane ograniczeniem z usuwaną rolą należy kolejno sprawdzić czy wyżej wymienione formuły należą do zbioru konsekwencji którejkolwiek z istniejących ról. Można to sprawdzić w prosty sposób, a mianowicie zadając po kolei wszystkim rolom pytania:

$$\Leftarrow \mathbf{POdgrywa}(P,Rola), \Leftarrow \mathbf{FOdgrywa}(P,Rola), \Leftarrow \mathbf{O}Odgrywa(P,Rola),$$

gdzie *Rola* jest nazwą usuwanej roli.

Po odnalezieniu ról powiązanych ograniczeniami z rolą usuwaną, należy usunąć z nich wyżej wymienione ograniczenia. Zabieg ten jest konieczny, bo o ile pozostawienie ograniczenia *wzajemnego wykluczania się* nie spowodowałoby negatywnych skutków, to pozostawienie formuły $\mathbf{O}Odgrywa(P,Rola)$, doprowadziłoby do dezorganizacji funkcjonowania systemu.

7.4.3. Asocjacja ról z podmiotami.

Jednym z podstawowych zadań podmiotów pełniących funkcje administracyjne jest tworzenie nowych powiązań pomiędzy rolami i podmiotami. Zakres kompetencji danego podmiotu administracyjnego jest wyznaczony dokładnie w ten sam sposób, w jaki wyznaczony jest zakres możliwości podmiotu w normalnej roli, tzn. jest on zdefiniowany jako zbiór konsekwencji logicznych zbioru formuł związanych z tą rolą.

Dodawanie do zbioru możliwości roli administracyjnej powinno tak samo jak w przypadku zwykłych ról, być poprzedzone operacją weryfikacji spójności nowego uprawnienia ze zbiorem już istniejących. Dodatkowym założeniem jakie należy poczynić w przypadku akcji przyznawania możliwości i wiązania podmiotów z rolami jest konieczność przeprowadzenia analizy spójności systemu po wykonanej każdorazowo operacji.

Przykładem formuły związanej z rolą administracyjną, która daje możliwość dodawania do ról użytkowników relacji dziedziczenia jest formuła prezentowana już wcześniej:

$$Poziom(R1, 15) \wedge Poziom_dziecka(R2, lev) \rightarrow \mathbf{PDodaj_dziedzicz}(R1,R2).$$

Podmioty administracyjne obok uprawnień do przyznawania/odbierania ról podmiotom, mogą posiadać również możliwość do delegowania ról, tzn. do krótkotrwałego przekazania danemu podmiotowi uprawnień związanych z rolą, z którą normalnie nie jest powiązany.

Można wyróżnić dwa rodzaje delegowania ról. Pierwszy z nich dotyczy przypadków, w których na czas aktywności w delegowanej roli podmiot traci normalnie przysługujące mu uprawnienia oraz drugi, kiedy podmiot zachowuje normalnie przysługujące mu uprawnienia oraz zyskuje nowe na skutek operacji delegacji.

Na poziomie definicji roli administracyjnej możliwość delegowania ról przybiera postać formuły określającej warunki delegacji i jej rodzaj np. w następujący sposób:

1. $Odgrywa(R, Admin) \rightarrow PDeleguj(R, Technik),$
2. $Odgrywa(R, Kierownik_tech) \rightarrow PDeleguj_B(R, Technik),$

gdzie:

$Deleguj(R, Technik)$ - zachowuje posiadane uprawnienia,

$Deleguj_B(R, Technik)$ - nie pozwala na zachowanie uprawnień.

Techniczne aspekty realizacji tych funkcji są umownie i zależne od konkretnego systemu, jego systemu operacyjnego, itp.

Rozważmy jeszcze jeden problem delegowania ról, a mianowicie wpływ tej operacji na możliwości podmiotów. Oczywistym wydaje się być fakt, że podmiot delegowany do odgrywania danej roli powinien dać się rozróżnić od podmiotu, który daną rolę odgrywa na zasadzie normalnej operacji przypisania. W wielu przypadkach konieczne jest również ograniczenie możliwości podmiotów delegowanych jedynie do kilku wybranych praw spośród całego zakresu przysługujących podmiotom w sposób stały związanych z daną rolą. Na gruncie zdefiniowanej metody opisu ról w *RMM* możemy łatwo uzyskać żądany efekt, poprzez dołączenie odpowiedniego warunku w formułach definiujących uprawnienia, np.:

F1: $\forall P, x \text{ Plik}(x) \wedge \text{Właściciel}(P, x) \rightarrow PCzytaj(P, x)$

definiujemy formułę:

F1': $\forall P, x \text{ Niedelegowany}(P) \wedge \text{Plik}(x) \wedge \text{Właściciel}(P, x) \rightarrow PCzytaj(P, x)$

Otrzymujemy w ten sposób ograniczenie możliwości delegowanego podmiotu w danej roli. W semantyce światów możliwych, ten zabieg będzie postrzegany jako ograniczenie relacji dostępności dla podmiotów delegowanych do odgrywania tej roli.

7.5. Podsumowanie.

W rozdziale siódmym w oparciu o opracowany wcześniej model *RMM* przedstawiony został opis mechanizmów zarządzania dostępem takich jak definiowanie powiązań typu rola-rola oraz powiązań rola-prawa dostępu i rola-podmiot.

Na potrzeby opisu wyżej wymienionych mechanizmów opracowano dwa algorytmy: badania spójności nowododawanej formuły z istniejącym zbiorem formuł i budowania hierarchii ról. Sformułowano i udowodniono również twierdzenia o spójności nowododawanej formuły z istniejącym zbiorem formuł, o ograniczeniu dziedziczenia uprawnień, o spójności hierarchii ról, o spójności hierarchii ról i o wzajemnym wymaganiu się ról.

8. Podsumowanie.

Niniejsza praca odnosi się do problemów związanych ze sterowaniem i zarządzaniem dostępem w systemach informatycznych. Zgodnie z zaleceniami definiowanymi przez instytucje zajmujące się określaniem wymagań związanych z ochroną zasobów systemów, zapewnienie wysokiego poziomu bezpieczeństwa systemu wymaga zdefiniowania odpowiedniego modelu. Właściwie skonstruowany model może pozwolić na dokładniejszą analizę aktualnego poziomu bezpieczeństwa systemu, jego weryfikację, a także na automatyzację niektórych procedur administracyjnych, związanych z zarządzaniem bezpieczeństwem. Aby można jednak uznać, że model reprezentuje w pożądanym stopniu zachowanie systemu rzeczywistego, konieczne jest przeprowadzenie dowodu odpowiedniości modelu dla danego systemu.

W związku ze złożonością problemu sterowania dostępem, wszystkie istniejące modele (np. cały zbiór rozwiązań opartych na macierzy dostępu) są w mniejszym lub większym stopniu uproszczeniami względem rzeczywistości. Jednym z możliwych kryteriów wyboru formalizmu na potrzeby modelowania systemu sterowania spośród rozmaitych dostępnych uproszczeń, może być możliwość operowania językiem charakterystycznym dla zarządzania wysokiego poziomu na poziomie sterowania dostępem. Samo wykorzystanie języka i pojęć zbliżonych do opisywanej rzeczywistości nie dowodzi jeszcze tego, że pozwala on modelować z większą precyzją stan rzeczywisty niż bardziej abstrakcyjny model. Niemniej jednak, zabieg taki pozwala na przeprowadzanie czynności związanych z administrowaniem systemem w sposób bardziej intuicyjny, a przez to umożliwia eliminację lub przynajmniej redukcję prawdopodobieństwa powstawania błędów na skutek pomyłek związanych z koniecznością manipulowania abstrakcyjnymi pojęciami przez osobę zarządzającą systemem.

Oprócz przybliżenia języka formalnego opisu systemu sterowania do języka naturalnego, kolejną pożądaną właściwością dla opisywanych mechanizmów sterowania, jest możliwość działania zgodnie z procedurami obowiązującymi na wyższym poziomie zarządzania bezpieczeństwem. Chodzi tu przede wszystkim o możliwość wykorzystania istniejących w każdej organizacji struktur obrazujących zależności pomiędzy komponentami systemu, polityk wysokiego poziomu, itp.

Dlatego też w pracy przyjęto następujący zbiór wymagań stanowiący punkt wyjścia dla opracowywanego modelu mechanizmu dla sterowania dostępem we współczesnych systemach informatycznych.

Założono, że model ten powinien wspierać politykę sterowania dostępem poprzez role (RBAC) oraz charakteryzować się następującymi właściwościami:

- Zgodnością z podstawowymi paradygmatami polityki kontroli dostępu poprzez role, tzn.:
 - intuicyjnością zapisu polityki,
 - możliwością zapisu polityk dla różnorodnych organizacji,

- Możliwością implementacji mechanizmów polityki kontroli dostępu poprzez role:
 - 1) definiowania ról (rola- {prawa,zakazy,obowiązki}),
 - 2) przypisywania podmiotów do ról (podmioty-role):
 - stałe,
 - delegowane,
 - 3) definiowania powiązań pomiędzy rolami (rola-rola):
 - hierarchii ról,
 - ograniczeń:
 - wzajemne wykluczanie się ról,
 - wzajemne wymaganie się ról,
 - ograniczenie na maksymalną liczbę aktywnych podmiotów przypisanych do roli.

- Możliwością automatyzacji implementacji mechanizmów polityki kontroli dostępu poprzez role.

W pracy zaproponowano model mechanizmów polityki sterowania i zarządzania dostępem poprzez role, a następnie przeprowadzono szczegółową analizę jego własności. Wśród najważniejszych cech opracowanego modelu należy wymienić:

- możliwość korzystania z translacji semi-funkcyjnej,
- intuicyjność zapisu polityki,
- możliwość zapisu polityk dla różnorodnych organizacji,
- możliwość formalnej analizy definiowanej polityki sterowania dostępem,
- możliwość korzystania z formalnej definicji pojęć takich jak: jest obowiązkowe, jest zabronione, jest dozwolone,
- możliwość rozstrzygania w sposób jednoznaczny czy dane żądanie jest zgodne z przyjętą polityką, czy pozostaje z nią w konflikcie,
- istnienie zdefiniowanego szeregu funkcji związanych z zarządzaniem dostępem,
- możliwość automatyzacji wielu elementów związanych z procesem zarządzania i sterowania dostępem.

Następnie przedstawiono szereg algorytmów pozwalających realizować sterowanie i zarządzanie dostępem oraz udowodniono twierdzenia o właściwościach tychże algorytmów. Przedstawiono opis z wykorzystaniem opracowanego modelu wyżej wymienionych mechanizmów dla polityki sterowania dostępem poprzez role i podano szereg ilustrujących jego możliwości przykładów.

Praca składa się z ośmiu rozdziałów. Pierwszy z nich stanowi wstęp, w którym zdefiniowano zakres tematyczny pracy oraz określono główny cel badawczy. W rozdziale drugim zawarte jest wprowadzenie w tematykę bezpieczeństwa systemów informatycznych. Zawarto w nim omówienie ogólnych pojęć charakterystycznych dla tego tematu oraz określone zostało w nim miejsce i rola jaką odgrywa we współczesnych systemach ochrony

sterowanie dostępem. Rozdział ten zawiera definicje podstawowych pojęć z zakresu bezpieczeństwa i sterowania dostępem. W kolejnym rozdziale (trzecim), przedstawiono charakterystykę trzech podstawowych polityk sterowania dostępem DAC, MAC oraz mechanizmów z nimi związanych. Rozdział czwarty poświęcony jest polityce RBAC. Obok ogólnej charakterystyki polityki sterowania dostępem poprzez role zawiera on krytyczną analizę istniejących implementacji systemów dla tej polityki. Przedstawiono w tym rozdziale szczegółowo kilka najważniejszych spośród istniejących mechanizmów, zwracając uwagę na wiążące się z nimi niedogodności oraz braki w stosunku do podstawowych założeń polityki RBAC i sterowania dostępem w ogóle. W rozdziale piątym na podstawie przeprowadzonej w dwóch poprzednich rozdziałach analizy, przedstawiono propozycję nowego modelu mechanizmów dla polityki RBAC. W tym celu przedstawiono krótkie uzasadnienie wykorzystania formalizmu logiki deontycznej na potrzeby opisu i analizy polityki RBAC. Następnie został zdefiniowany formalny język, który stanowi podstawę opracowanego modelu oraz zostały określone zasady korzystania z niego na potrzeby sterowania i zarządzania dostępem. W rozdziale tym udowodniono właściwości modelu *RMM* i określono jego moc ekspresji. Rozdział szósty poświęcony jest mechanizmowi sterowania dostępem. Zawiera on algorytm analizy spójności żądań podmiotów z przyjętą polityką oraz dowód twierdzenia o dopuszczeniu do wykonania przez podmiot operacji dostępu. W rozdziale siódmy przedstawiono opis w modelu *RMM* mechanizmów zarządzania dostępem. Znajdują się w nim dwa opracowane algorytmy: badania spójności nowododawanej formuły z istniejącym zbiorem formuł i budowania hierarchii ról oraz twierdzenia o spójności nowododawanej formuły z istniejącym zbiorem formuł, o ograniczeniu dziedziczenia uprawnień, o spójności hierarchii ról, o spójności hierarchii ról i o wzajemnym wymaganiu się ról. Ostatni, ósmy rozdział stanowi podsumowanie całej pracy.

Najważniejsze rezultaty otrzymane w pracy w poszczególnych jej rozdziałach:

Rozdział 3. Przegląd podstawowych polityk sterowania dostępem.

- Krytyczna analiza możliwości i zakresu potencjalnych zastosowań we współczesnych systemach informatycznych dla trzech podstawowych polityk sterowania dostępem.

Rozdział 4. Polityka i mechanizmy RBAC.

- Krytyczna analiza możliwości realizacji zadań związanych ze sterowaniem dostępem poprzez role przez przedstawione implementacje,
- Sformułowanie, na podstawie wcześniejszej analizy oraz prezentacji polityk zbioru założeń i wymagań niezbędnych dla uzyskania efektywnego modelu dla mechanizmu polityki RBAC,
- Wykazanie konieczności kontrolowania spójności uregulowań jako jednego z podstawowych problemów na drodze do automatyzacji zarządzania RBAC.

Rozdział 5. Model sterowania i zarządzania dostępem oparty o politykę RBAC.

- Definicja modelu RMM,
- Określenie cech funkcjonalnych proponowanego modelu
- Dowód możliwości zastosowania metod automatycznego wnioskowania dla opracowanego modelu.
- Opracowanie algorytmów:
 - sprawdzenia czy formuła języka *RMM* jest w postaci koniunkcyjnej klauzul Horna,
- Udowodnienie twierdzeń:
 - o możliwości translacji formuł języka *RMM* do postaci koniunkcyjnej klauzul Horna,
 - o możliwości sprowadzenia reguły języka *RMM* do postaci *kkH*
- Sformułowanie metody automatyzacji wybranych elementów procesu sterowania i zarządzania dostępem w oparciu o formalizm logiki deontycznej oraz programy logiczne języka PROLOG,

Rozdz. 6 - Mechanizm sterowania dostępem

- Opracowanie algorytmu:
 - analizy spójności żądań dostępu z przyjętą polityką,
- Udowodnienie twierdzenia:
 - o dopuszczeniu do realizacji przez podmiot żądania dostępu,

Rozdz. 7 - Mechanizmy zarządzania dostępem.

- Przedstawienie sposobu zarządzania dostępem w oparciu o opracowany model,
- Opis mechanizmów definiowania powiązań: rola-prawa dostępu, rola-rola, rola-podmiot, dla polityki kontroli dostępu poprzez role
- Opracowanie metod pozwalających na dowodzenie spójności przyjętych uregulowań (w obrębie danej roli jak i całego systemu norm związanych z danym systemem),
- Opracowanie algorytmów:
 - badania spójności nowo dodawanej formuły z istniejącym zbiorem formuł,
 - budowania hierarchii ról.
- Udowodnienie twierdzeń:
 - o spójności nowo dodawanej formuły z istniejącym zbiorem formuł,
 - o ograniczeniu dziedziczenia uprawnień,
 - o spójności hierarchii ról,
 - o wzajemnym wykluczaniu się ról,
 - o wzajemnym wymaganiu się ról.

W związku z powyższym możemy stwierdzić, że zaproponowany model sterowania dostępem poprzez role jest realizacją postawionego w pierwszym rozdziale głównego problemu badawczego niniejszej pracy. Niemniej jednak model ten zapewne nie stanowi rozwiązania wszystkich problemów związanych ze sterowaniem dostępem w polityce RBAC. Aby dokonać rzetelnej jego oceny, należałoby przeprowadzić szereg dodatkowych badań, umożliwiających praktyczną weryfikację teoretycznie wykazanych możliwości oraz

porównanie otrzymanych rezultatów z innymi istniejącymi rozwiązaniami. Wśród zadań, mogących stanowić punkt wyjścia przy kontynuacji przeprowadzonych do tej pory badań, można wymienić:

- określenie zależności pomiędzy wzrostem ilości definiowanych norm, a zapotrzebowaniem na czas w procesie weryfikacji ich spójności,
- bardziej szczegółowa analiza możliwości i ewentualnych ograniczeń związanych z zastosowaniem zdefiniowanego mechanizmu do sterowania operacjami (transakcjami) w systemie,
- wprowadzenie priorytetów ról,
- badanie konsekwencji wykroczenia przeciw obowiązującym w systemie normom (np. możliwość definiowania sankcji wobec podmiotów naruszających bezpieczeństwo systemu, definicja sposobu restytucji stanu bezpieczeństwa, itp.).

Odrębnym problemem mogącym stanowić interesujący przedmiot badań jest przeprowadzenie pełnej analizy semantycznej zagadnień związanych z sterowaniem dostępem oraz możliwościami modelowania ich w logice deontycznej.

Przedstawiane w niniejszej pracy rozważania koncentrowały się wokół zagadnień związanych ze sterowaniem dostępem do systemów informatycznych na poziomie logicznym. Jednym z podstawowych wymagań, jakie stawia się przed każdym systemem jest możliwość przewidywania jego reakcji oraz powtarzalność zachowania. Dąży się do uzyskania takiego efektu zarówno na drodze stosowania nowych technologii pozwalających na zwiększenie niezawodności sprzętu wchodzącego w skład systemu, jak i na drodze opracowywania coraz to bardziej niezawodnych procedur na poziomie logicznym funkcjonowania systemu. W związku z tym, również i od logicznego sterowania dostępem wymaga się, aby było możliwe formalne wykazanie jego możliwości. Ta konieczność, stanowi główny motyw podjętych prac nad nowym modelem sterowania dostępem. Ponieważ większość współczesnych zastosowań systemów informatycznych kierowana jest do tzw. organizacji komercyjnych, jako podstawę dla tworzonego mechanizmu wybrano politykę RBAC, gdyż właśnie ta polityka jest w sposób szczególnie dedykowana takiemu kręgowi odbiorców. Drugą ideą związaną z podjętą pracą, było umożliwienie intuicyjnego przenoszenia polityk wysokiego poziomu, formułowanych w języku naturalnym, na poziom sterowania dostępem. Przedstawiony w poprzedniej części tej pracy mechanizm oparty jest na formalizmie logiki deontycznej, która z jednej strony daje możliwość przeprowadzenia w sposób ścisły wnioskowania o prawdziwości formułowanych w niej twierdzeń, a z drugiej strony daje możliwość operowania językiem na stosunkowo wysokim poziomie abstrakcji. Na podstawie przedstawionego w poprzednich rozdziałach materiału można uznać główną tezę pracy za udowodnioną.

Dodatek A. Wykaz najważniejszych skrótów występujących w pracy.

CC	- Common Criteria
DAC	- Discretionary Access Control (uznaniowe sterowanie dostępem)
DoD	- Department of Defence
FOL	- First Order Logic (klasyczna logika pierwszego zędu)
LAC	- Logical Access Control (logiczne sterowanie dostępem)
MAC	- Mandatory Access Control (narzucone sterowanie dostępem)
NIST	- National Institute of Standards and Technology
RBAC	- Role Based Access Control (sterowanie dostępem poprzez role)
RMM	- RBAC Modal Model
SI	- System informatyczny
q.e.d.	- łac. <i>quod erat demonstrandum</i> - to co było do udowodnienia

Dodatek B. Pomocnicze definicje z zakresu matematyki i logiki.

Symbole matematyczne:

$Cn(X)$	- zbiór wszystkich konsekwencji zbioru X
$card(X)$	- liczebność zbioru X
$Rez(X)$	- zbiór rezolwent zbioru X
$\Leftrightarrow$	- równoważność (wtedy i tylko wtedy, gdy)
kkH	- postać koniunkcyjna klauzul Horna
PNF	- postać normalna prefiksowa,
CNF	- postać normalna koniunkcyjna,
$Skol(f)$	- postać formuły po skolemizacji
$O\varphi$	- φ jest obowiązkowe,
$P\varphi$	- φ jest dozwolone,
$F\varphi$	- φ jest zabronione.
$T, TRUE$	- wartość logiczna - prawda ,
$F, FALSE$	- wartość logiczna - fałsz

Definicje:

Definicja B.1. *Translacja semi-funkcyjna.*

Definicja B.2. *Definicje pojęć literału i klauzuli.*

Definicja B.3. *Definicja pojęcia rezolwenty.*

Definicja B.4. *Formuły w postaci przedrostkowej normalnej.*

Definicja B.5. *Pojęcie dowodu.*

Definicja B.6 *Pojęcie logicznej konsekwencji.*

Definicja B.1. *Translacja semi-funkcyjna.*

Przez translację semi-funkcyjną logiki multimodalnej o zbiorze modalności $\{ \Diamond_m \}_{m \in M}$, oznaczaną przez T_{sf} , rozumiemy odwzorowanie przyporządkowujące formułom multimodalnym i światom formuły logiki klasycznej FOL , spełniające następujące warunki :

$T_{sf}(\phi, x) = P(x)$, gdzie ϕ jest zdaniem atomowym, zaś P - odpowiadającym mu predykatem;

$T_{sf}(\Diamond_m \phi, x) = \forall y [R_m(x, y) \rightarrow T_{sf}(\phi, y)]$

$T_{sf}(\Diamond_m \phi, x) = \exists f: AF_m[def(f(x)) \wedge T_{sf}(\phi, f(x))]$,

gdzie :

AF_m jest zbiorem funkcji odpowiadających relacji dostępności światów dla modalności $\Diamond_m$

$def(f(x))$ jest predykatem oznaczającym, iż wartość $f(x)$ jest określona (dla logik serialnych $def(f(x)) \equiv T$),

T_{sf} zachowuje spójniki klasyczne i kwantyfikatory oraz jest zdefiniowane dla języka pierwszego rzędu w następujący sposób:

$T_{sf}(\neg \phi, x) = \neg T_{sf}(\phi, x)$, $T_{sf}(\phi \vee \varphi, x) = T_{sf}(\phi, x) \vee T_{sf}(\varphi, x)$, $T_{sf}(\phi \wedge \varphi, x) = T_{sf}(\phi, x) \wedge T_{sf}(\varphi, x)$, $T_{sf}(\phi \rightarrow \varphi, x) = T_{sf}(\phi, x) \rightarrow T_{sf}(\varphi, x)$, $T_{sf}(\phi \leftrightarrow \varphi, x) = T_{sf}(\phi, x) \leftrightarrow T_{sf}(\varphi, x)$,

$T_{sf}(\forall u \phi, x) = \forall u T_{sf}(\phi, x)$, $T_{sf}(\exists u \phi, x) = \exists u T_{sf}(\phi, x)$

dla każdego m-argumentowego predykatu r i termów $t_1, \dots, t_m$, mamy:

$T_{sf}(r(t_1, \dots, t_m), x) = r(x, t_1, \dots, t_m)$

Definicja B.2. *Definicje pojęć literału i klauzuli.*

Niech X będzie formułą bez kwantyfikatorów (tzn. formułą otwartą).

atom - $p(t_1, \dots, t_n)$ gdzie p jest predykatem, zaś $t_1, \dots, t_n$ są termami

literal - a albo $\neg a$, gdzie a jest atomem

literal pozytywny - a , gdzie a jest atomem

literal negatywny - $\neg a$, gdzie a jest atomem

literały komplementarne - a oraz $\neg a$

klauzula - $l_1 \vee l_2 \vee \dots \vee l_n$ gdzie $l_1, l_2, \dots, l_n$ są literałami ($n \geq 0$)

klauzula Horna - klauzula która ma co najwyżej jeden literal pozytywny

Definicja B.3. *Definicja pojęcia rezolwenty.*

Niech C_1, C_2 będą dwoma klauzulami, które zawierają literały komplementarne L_1, L_2 . Przez $(C_1 \setminus L_1)$ będziemy oznaczać klauzulę C_1 , z której usunięto literał L_1 . Klauzulę $(C_1 \setminus L_1) \vee (C_2 \setminus L_2)$ nazywamy *rezolwentą*. Reguła rezolucji:

$$\frac{C_1, C_2}{(C_1 \setminus L_1) \vee (C_2 \setminus L_2)} \quad L_1 \in C_1, L_2 \in C_2, L_1, L_2 \text{ są komplementarne}$$

Definicja B.4. *Formuły w postaci przedrostkowej normalnej.*

Formuła znajduje się w postaci *przedrostkowej normalnej* (PNF - *prenex normal form*) wtt, gdy jest ona w postaci

$$Q_1 x_1 \cdot Q_2 x_2 \dots Q_n x_n \cdot X$$

gdzie $Q_1, Q_2, \dots, Q_n \in \{\forall, \exists\}$, zaś formuła X nie zawiera kwantyfikatorów i zawiera spójniki logiczne negacji, koniunkcji i alternatywy.

Część

$$Q_1 x_1 \cdot Q_2 x_2 \dots Q_n x_n \cdot$$

nazywamy *przedrostkiem*, a

$$X$$

nazywamy *matrycą* formuły w postaci PNF.

Definicja B.5. *Pojęcie dowodu.*

Dowodem formuły zdaniowej A w oparciu o zbiór X formuł zdaniowych nazywamy każdy skończony ciąg formuł $D_1, D_2, D_3, \dots, D_n$ taki, że $D_n = A$ oraz dla każdego wskaźnika $k \leq n$ spełniony jest przynajmniej jeden z następujących warunków:

- $D_k \in X$;
- istnieją: wskaźniki $i < k$, term α oraz wskaźniki j takie, że α jest podstawialne za x_j do D_j oraz $D_k = S(\alpha, x_j, D_j)$;
- istnieją takie wskaźniki $i, j < k$, że $D_i = \lceil D_j \rightarrow D_k \rceil$;
- istnieją: wskaźniki $i < k$, formuły zdaniowe B i C oraz wskaźnik j takie, że $D_i = \lceil B \rightarrow \forall x_j C \rceil$ oraz $D_k = \lceil B \rightarrow C \rceil$;
- istnieją: wskaźniki $i < k$, formuły zdaniowe B i C oraz wskaźnik j takie, że zmienna x_j nie jest wolna w B , $D_i = \lceil B \rightarrow C \rceil$ oraz $D_k = \lceil B \rightarrow \forall x_j C \rceil$;
- istnieją: wskaźniki $i < k$, formuły zdaniowe B i C oraz wskaźnik j takie, że $D_i = \lceil \exists x_j B \rightarrow C \rceil$ oraz $D_k = \lceil B \rightarrow C \rceil$;
- istnieją: wskaźniki $i < k$, formuły zdaniowe B i C oraz wskaźnik j takie, że zmienna x_j nie jest wolna w B , $D_i = \lceil B \rightarrow C \rceil$ oraz $D_k = \lceil \exists x_j B \rightarrow C \rceil$.

Definicja B.6 *Pojęcie logicznej konsekwencji.*

Formuła zdaniowa A jest konsekwencją zbioru X formuł zdaniowych wtedy i tylko wtedy, gdy istnieje przynajmniej jeden dowód formuły A w oparciu o zbiór X . Zbiór wszystkich konsekwencji zbioru X oznacza się $Cn(X)$.

Twierdzenia:

Twierdzenie B.1. *O własności rezolwenty.*

Twierdzenie B.2. *O spełnialności formuł w postaci Skolema*

Twierdzenie B.1. *O własności rezolwenty.*

Rezolwenta jest logiczną konsekwencją klauzul C_1, C_2 .

Twierdzenie B.2. *O spełnialności formuł w postaci Skolema*

Formuła X jest spełnialna wtt, gdy spełnialna jest formuła $Skol(X)$.

Algorytmy:

Algorytm B.1. *Badanie niespełnialności zbioru klauzul*

Algorytm B.2. *Algorytm skolemizacji formuł.*

Algorytm B.3. *Algorytm Sprowadzania formuł otwartych do koniunkcyjnej postaci normalnej.*

Algorytm B.4. *Algorytm sprowadzania do przedrostkowej postaci normalnej.*

Algorytm B.1. *Badanie niespełnialności zbioru klauzul*

Dane: zbiór klauzul S

Wynik: odpowiedź tak/nie

Procedura

while

- $\notin S$ i istnieją $C1, C2 \in S$ dające rezolwentę nie należącą do S

do

1. znajdź $C1, C2$ które dają się uzgodnić i wyliczyć ich rezolwentę
2. zastąpić S przez S z dołączoną rezolwentą

od

Jeżeli S zawiera klauzulę pustą - odpowiedź tak, w przypadku przeciwnym - nie.

end.

Algorytm B.2. *Algorytm skolemizacji formuł.*

Dane: formuła X

Wynik: formuła $Skol(X)$, równoważna w sensie *spełnialności* formule X, w postaci PNF, bez kwantyfikatorów egzystencjalnych. Równoważność w sensie spełnialności oznacza, że $Skol(X)$ jest spełnialna wtt, gdy X jest spełnialna

Procedura:

1. Niech Y będzie formułą PNF(X) - wynik poprzedniego algorytmu.

2. **while**

istnieje w przedrostku formuły formuły Y kwantyfikator egzystencjalny

do

- (a) jeżeli Y ma postać:

$$\exists x. Z(x, y, \dots),$$

to zastąp ją formułą postaci

$$Z(c, y, \dots)$$

gdzie c jest nową stałą (nowym symbolem stałej)

- (b) w przeciwnym przypadku niech Y ma postać:

$$\forall x_1. \forall x_2. \dots \forall x_m. \exists y. Z(x_1, x_2, \dots, x_m, y, \dots)$$

wtedy zastąp ją formułą postaci:

$$\forall x_1. \forall x_2. \dots \forall x_m. Z(x_1, x_2, \dots, x_m, f(x_1, x_2, \dots, x_m), \dots)$$

gdzie f jest nowym funktorem

od

3. Formułę Y definiujemy jako $Skol(X)$

end.

Algorytm B.3. *Algorytm Srowadzania formuł otwartych do koniunkcyjnej postaci normalnej.*

Dane: formuła X

Wynik: formuła CNF(X), równoważna formule X

Procedura:

1. Niech Y będzie początkowo formułą X, w której występują tylko spójniki koniunkcji, dysjunkcji i negacji.

2. **while**

Y nie jest w postaci koniunkcyjnej normalnej

do

zastępuj podformuły formuły Y zgodnie z regułami:

formuła zastępowana	formuła zastępująca
$\neg(\neg A)$	A
$\neg(A \vee B)$	$\neg A \wedge \neg B$
$\neg(A \wedge B)$	$\neg A \vee \neg B$
$A \vee (B \wedge C)$	$(A \vee B) \wedge (A \vee C)$

od

Formułę Y definiujemy jako CNF(X)

end.

Algorytm B.4. *Algorytm sprowadzania do przedrostkowej postaci normalnej.*

Dane: formuła X

Wynik: formuła PNF(X), równoważna *semantycznie* formule X, i będąca w postaci PNF

Procedura: Polega na etapowym przekształcaniu formuły X. Formuła pośrednia jest oznaczana przez Y.

1. Przyjmij Y jest równoważna X.

2. Wyeliminuj spójniki logiczne: $\Rightarrow$, $\Leftrightarrow$ (wg zasad przedstawionych poprzednio).

3. Wprowadź znak negacji bezpośrednio przed symbole predykatowe zastępując (dopóki można) podformuły zgodnie z poniższą tabelą:

Formuła zastępowana	Formuła zastępująca
$\neg(\neg A)$	A
$\neg(A \vee B)$	$\neg A \wedge \neg B$
$\neg(A \wedge B)$	$\neg A \vee \neg B$
$\neg \forall x. A$	$\exists x. \neg A$
$\neg \exists x. A$	$\forall x. \neg A$

4. Dopóki formuła Y nie jest w postaci PNF, przekształcaj ją zgodnie z następującymi regułami, w których $Q, Q_1, Q_2 \in \{\forall, \exists\}$, oraz - w przypadku dwóch ostatnich reguł - z jest zmienną nie występującą w A :

Formuła zastępowana	Formuła zastępująca
$Qx.A(x) \vee B$	$Qx.(A(x) \vee B)$
$Qx.A(x) \wedge B$	$Qx.(A(x) \wedge B)$
$\forall x.A(x) \wedge \forall x.B(x)$	$\forall x.(A(x) \wedge B(x))$
$\exists x.A(x) \vee \exists x.B(x)$	$\exists x.(A(x) \vee \exists x.B(x))$
$Q_1x.A(x) \vee Q_2y.B(y)$	$Q_1x.Q_2z.(A(x) \vee B(z))$
$Q_1x.A(x) \wedge Q_2y.B(y)$	$Q_1x.Q_2z.(A(x) \wedge B(z))$

- Uwagi* Stosowany tu zapis $A(x)$ oznacza, że x jest zmienną, która ma wolne wystąpienia w formule A .
Zmienne wiązane przez kwantyfikatory mają znaczenie lokalne. Dlatego jeżeli w formule B występuje zmienna x , to zmienna x w formule A należy przemianować na dowolną inną zmienną, która w B nie występuje.
5. Formułę PNF(X) definiujemy jako formułę Y .
end.

Dodatek C. Elementy opracowanego modelu sterowania dostępem.

Syntaktyka języka *RMM* (*RBAC Modal Model*).

Słownik języka *RMM* składa się z następujących rodzajów wyrażeń:

a) zmiennych zdaniowych:

$p \ q \ r \ p_1 \ q_1 \dots$

b) zmiennych indywiduowych:

$x \ y \ z \ x_1 \ y_1 \ z_1 \dots$

c) n -argumentowych stałych predykatowych ($n=1,2,3,\dots$):

$Poziom(x,y), \quad Utworz(x,y), Usun(x), Poziom_dziecka(x, \quad y), \quad Dodaj_dziedzicz(x,y),$
 $Grupa(x,y), Deleguj(x,y), Deleguj_B(x,y)$
 $Podmiot(x), Rola(x), Plik(x), Usun(x), Przyznaj_role(x,y), Zakres(x, y,z)$
 $Odgrywa(x, y), Dzień(x), Tworz_kopie(x), Dziedziczy(x), Zawiera_sie(x,y,z), \dots$

d) stałych logicznych (spójniki logiczne i kwantyfikatory):

$\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \forall, \exists$

e) operatorów deontycznych:

P, O, F,

odczytywanych w następujący sposób:

O ϕ - ϕ jest obowiązkowe,

P ϕ - ϕ jest dozwolone,

F ϕ - ϕ jest zabronione.

f) równoważności określających zależności pomiędzy operatorami deontycznymi zdefiniowanej przez nas logiki:

- **O** $\phi \leftrightarrow \neg \mathbf{P} \neg \phi$,
- **P** $\phi \leftrightarrow \neg \mathbf{F} \phi$.
-

g) nawiasów użytych jako znaków przystankowych

Reguły formowania języka *RMM*.

Formułą języka *RMM* jest:

- a) formuła atomowa, tzn. n -argumentowy predykat z n -krotnym wystąpieniem w nim zmiennych indywiduowych i/lub stałych indywiduowych,
- b) negacja formuły,

- c) implikacja, koniunkcja lub alternatywa utworzona z dowolnych dwóch formuł,
- d) wyrażenie powstałe z poprzedzenia formuły kwantyfikatorem,
- e) jeśli ϕ jest formułą, to także $\mathbf{O}\phi$, $\mathbf{P}\phi$ i $\mathbf{F}\phi$ jest formułą,

Aksjomaty *RMM*.

Do aksjomatów języka *RMM* należą:

- a) wszystkie aksjomaty *FOL*,
- b) aksjomat zdaniowej logiki deontycznej:

$$D = \mathbf{O}\phi \rightarrow \mathbf{P}\phi$$

Reguły inferencyjne.

Do reguł inferencyjnych języka *RMM* należą wszystkie reguły z *FOL*.

Semantyka języka *RMM* (*RBAC Modal Language*).

Niech model Kripkego dla logiki *RMM* będzie opisany przez trójkę [Kri63]:

$$\langle K, w_0, v, \rangle$$

gdzie:

- $K = \langle W, R, \rangle$ - jest s.strukturą Kripkego, gdzie:
- W - dowolny zbiór (tzw. światów możliwych),
 - R - relacja binarna określająca istniejące powiązania pomiędzy światami możliwymi,

(zakładam, że interpretacja funkcji i relacji języka nie zmienia się w zależności od aktualnego świata),

- Z - jest zbiorem formuł języka *RMM*,
- v - jest funkcją przyporządkowującą wartości logiczne zdaniom atomowym w światach $v: Z \times W \rightarrow \{T, F\}$ (wartości logiczne $T \equiv$ prawda, $F \equiv$ fałsz)

W takim przypadku, logiką modalną pierwszego rzędu o modalności $\mathbf{O}$ nazwiemy logikę $M = \langle F, C, \models_M \rangle$, gdzie:

- C - jest pewną podklasą klasy modeli Kripkego,
- $\models_M$ - relacja zdefiniowana w następujący sposób:
 - $K, w, v, \models_m \phi \Leftrightarrow v(\phi, w) = T$,
 - $K, w, v, \models_m \mathbf{O}\phi \Leftrightarrow$ dla każdego w' takiego, że $R(w, w')$ mamy $v(\phi, w') = T$,
 - $K, w, v, \models_m \mathbf{P}\phi \Leftrightarrow$ istnieje takie w' , że $R(w, w')$ mamy $v(\phi, w') = T$

Interpretacja elementów semantyki języka *RMM*.

Elementom struktury Kripkego nadaję następującą interpretację:

- W - zbiór światów możliwych, będzie interpretowany jako zbiór możliwych stanów aktywności systemu. Zakładam, przy tym, że stan systemu da się jednoznacznie opisać poprzez określenie stanów dla wszystkich elementów aktywnych w systemie (podmiotów) oraz wartości zdefiniowanych w systemie zmiennych globalnych
- $\forall w \in W$ - każdy ze światów możliwych będzie interpretowany jako aktualny stan aktywności określonego podmiotu w systemie,
- R - relacja wiążąca światy możliwe będzie interpretowana jako funkcja przejścia pomiędzy poszczególnymi stanami aktywności podmiotów.

Znaczenie najważniejszych predykatów języka *RMM*:

Przyjmuję następującą konwencję definiowania znaczenia predykatów:

$\text{Predykat}(x,y)$

$T \Leftrightarrow x = \text{ktoś} \text{ i } y = \text{coś},$

oznacza, że $\text{Predykat}(x,y)$ przyjmuje wartość T , wtedy i tylko wtedy, gdy zmienna $x = \text{ktoś}$, a zmienna $y = \text{coś}$.

$\text{Deleguj}(R,R2)$

$T \Leftrightarrow$ podmiot R jest delegowany z zachowaniem posiadanych uprawnień do roli $R2$

$\text{Deleguj}_B(R, \text{Technik})$

$T \Leftrightarrow$ podmiot R jest delegowany z bez zachowania posiadanych uprawnień do roli $R2$

$\text{Dodaj_dziedzicz}(R1,R2).$

$T \Leftrightarrow$ podmiot aktywny dodaje dziedziczenie roli $R2$ do zbioru norm roli $R1$

$\text{Dziedziczy}(R);$

$T \Leftrightarrow$ rola bieżąca dziedziczy rolę R

$\text{Dzień}(x)$

$T \Leftrightarrow x$ jest nazwą dnia tygodnia,

$\text{Grupa}(R,gr)$

$T \Leftrightarrow$ rola R należy do grupy gr

$\text{Odgrywa}(Pd, R)$

$T \Leftrightarrow$ podmiot Pd odgrywa rolę R

$\text{Plik}(x)$

$T \Leftrightarrow x$ jest nazwą pliku,

$\text{Podmiot}(x)$

$T \Leftrightarrow x$ jest nazwą podmiotu,

$\text{Poziom}(R, lev)$

$T \Leftrightarrow$ jeżeli rola R należy do poziomu klasyfikacji równego lev

Poziom_dziecka(R, lev)

$T \Leftrightarrow$

Przyznaj_rolę(Pd, R)

$T \Leftrightarrow$ podmiot administracyjny przyznaje rolę R podmiotowi Pd

Rola(x)

$T \Leftrightarrow$ x jest nazwą roli

Twórz_kopię(x)

$T \Leftrightarrow$ podmiot tworzy kopię bezpieczeństwa zasobu określonego przez nazwę x

Utwórz(R,lev);

$T \Leftrightarrow$ podmiot administracyjny tworzy nową rolę R na poziomie lev

Usuń(x);

$T \Leftrightarrow$ podmiot usuwa x

Zakres(x, y, z)

$T \Leftrightarrow$ jeżeli x jest w hierarchii ról pomiędzy poziomem y, a z

Dodatek D. Bibliografia.

- [ABN89] Abadi M., Burrows M., Needham R., *A logic of authentication*, Technical Report SRC Resaerch Report 39, Digital Equipment Corporation, 1989.
- [Bar95a] Barkley J., *Application Engineering in Health Care*, Proceedings of the 2nd Annual CHIN Summit, 1995.
- [Bar95b] Barkley J., *Implementing Role-Based Access Control using Object Technology*, First ACM Workshop on Role-Based Access Control, Gaithersburg, Maryland, November 30-December 1, 1995.
- [Bar97] Barkley J., *Comparing Simple Role Based Access Control Models and Access Control Lists* Second ACM Workshop on RBAC, 1997
- [Bat94] Batóg T., *Podstawy logiki*, Wydawnictwo Naukowe UAM, Poznań 1994.
- [BC91a] Bieber P., Cuppens F., *A Definition of Secure Dependencies Using the Logic of Security*. In proc. of The Computer Security Foundations Workshop, Franconia, 1991
- [BC91b] Bieber P., Cuppens F., *Computer Security Policies, Deontic Logic*. In Proc.of the First International Workshop on Deontic Logic in Computer Science, Amsterdam, The Netherlands, 1991.
- [BC92] Bieber P., Cuppens F. *A Logical View of Secure Dependencies*. Journal of Computer Security, 1(1):99--129, 1992.
- [BC93] Bieber P., Cuppens F., *Expression of confidentiality policies with deontic logic*. In Deontic Logic in Computer Science. Normative System Specification pp. (103-123), Chichester, John Wiley & Sons, 1993
- [BCFGK97] Barkley, Cincotta, Ferraiolo, Gavrilla, Kuhn *Role based Access Control for the World Wide Web*. Proceedings of the NIST-NSA National Computer Security Conference Oct. 1997,
- [BDRS95] Bolc L., Dziewicki K., Rychlik P., Szałas A., *Wnioskowanie w logikach nieklasycznych. Podstawy teoretyczne*. Akademicka Oficyna Wydawnicza PLJ, Warszawa 1995.
- [BDRS98] Bolc L., Dziewicki K., Rychlik P., Szałas A., *Wnioskowanie w logikach nieklasycznych. Automatyzacja wnioskowania*. Akademicka Oficyna Wydawnicza PLJ, Warszawa 1998.
- [Bib77] Biba K.J., *Integrity Considerations for Secure Computer Systems*, Tech. Report ESD-TR-76-372, USAF Electronic Systems Division, Bedford, USA, 1977.
- [BK85] Boebert W., Kain R., *A practical alternative to hierarchical integrity policies*. In NBSNCSC National Computer Security Conference, pages 18 -27, 1985.
- [BN89] Brewer D.F.C., Nash M.J., *The Chinese wall security policy*. In Proceedings IEEE Computer Society Symposium on Security and Privacy, pages 215-228, Oakland, CA, May 1989.
- [Bor77] Borkowski L., *Logika formalna*, PWN, Warszawa 1977

- [Car83] McCarty L.. *Permissions and obligations*. In pros. of the 8th Int. Joint. Conf. on AI, pp. 287-294, Karlsruhe, Germany, 1983
- [CC95] Cholvy L., Cuppens F. *Solving normative conflicts by merging roles*. In Fifth International Conference on Artificial Intelligence, Law, University of Maryland, USA, 1995.
- [CC97] Cholvy L., Cuppens F., *Analyzing Consistency of Security Policies*, In IEEE Symposium on Security, Privacy, Oakland, 1997.
- [CC98] Cholvy L., Cuppens F., *Reasoning about norms provided by conflicting regulations*, in Proceedings of the Fourth International Workshop on Deontic Logic in Computer Science, 1998.
- [CD96] Cuppens F., Demolombe R. *A Deontic Logic for Reasoning about Confidentiality*. In M.Brown, J.Carmo, editors, Deontic Logic, Agency, Normative Systems, Workshops in Computing. Springer, 1996.
- [Cho97] Cholvy L., *An application of SOL-resolution: checking regulation consistency*, In Collection of poster abstracts, IJCAI 1997
- [CORBA95] CORBA Security Draft, Object Management Group (OMG) Document Number 95-9-1, September 1995.
- [ChS96] Chen F., Sandhu R.S., *Constraints for Role-Based Access Control*, ACM Workshop, MD, USA, 1996.
- [CS96] Cuppens F., Saurel C. *Specyfing a Security Policy: A Case Study.*, In Proc. of the computer security foundations workshop, Kenmare, Co. Kerry, Ireland 1996.
- [Cup93] Cuppens F. *A Logical Analysis of Authorized, Prohibited Information Flows*. In IEEE Symposium on Security, Privacy, Oakland, 1993.
- [Cup94] Cuppens F. *Roles and Deontic Logic*. In A.J. I.Jones and M.Sergot, editors, Second International Workshop on Deontic Logic in Computer Science, Oslo, Norway, 1994.
- [Cup95] Cuppens F., *A Normative Framework for Security Policies*,
- [CW87] Clark D., Wilson D.R. *A Comparison of Commercial and Military Computer Security Policies*, 8th IEEE Symposium on Security and Privacy, Oakland, California, USA, 1987.
- [Den74] Denning D.E. *A Lattice Model of Secure Information Flow*, ACM Communications 19(5), May 1976, pp. 236-243.
- [Den76] Denning D.E. *On the Derivation of Lattice structured Information Flow Policies*, CSD TR 180, Comp. Sciences Dept. Purdue University, Lafayette, March 1976.
- [Den77] Denning D.E., Denning P.J., *Certification of Programs for Secure Information Flow*, ACM Communications, 20(7), July 1977, pp. 504-513.
- [Den82] Denning D.E. *Cryptography and Data Security*, Addison-Wesley Publishing, 1982.
- [Den86] Denning D.E., Akl S.G., Morgenstern M., *Views for Multilevel Database Security*, IEEE Comp. Security Symposium on Security and Privacy, Oakland, USA, 1986, pp. 156-172
- [FB97] Ferraiolo D.F., Barkley F., *Specifying and Managing Role-Based Access Control within a Corporate Intranet*, Second ACM Workshop on RBAC, 1997

- [FBK95] Ferraiolo D.F., Barkley F. Kuhn D.R., *A Role-Based Access Control Model and Reference Implementation within a Corporate Intranet*, First ACM Workshop on RBAC, 1995
- [FCK95] Ferraiolo D.F. Cugini J., Kuhn D.R. , *Role-Based Access Control (RBAC): Features and Motivations*, 11th Annual Computer Security Applications Proceedings, 1995.
- [FGL93] Ferraiolo D.F., Gilbert D., Lynch N., *An Examination of Federal and commercial Access Control Policy Needs*, Proceedings of the 16th NIST-NSA National Computer Security Conference, Baltimore, Maryland, September 20-23, 1993.
- [FK92] Ferraiolo D.F., Kuhn D.R., *Role-Based Access Controls*, Proceedings of the 15th NIST-NSA National Computer Security Conference, Baltimore, Maryland, October 13-16, 1992.
- [FK95] Ferraiolo D.F., Kuhn D.R., *Future Directions in Role-Based Access Control* First ACM Workshop on RBAC, 1995
- [FT98] Firozabadi B.S., Tan Y.H., *Formal models of fraud*, in Proceedings of the Fourth International Workshop on Deontic Logic in Computer Science, pp. 371--385, 1998.
- [GC95] Griew A., Currell R., *A Strategy for Security of the Electronic Patient Record*, Institute for Health Informatics, Aberystwyth, Draft Version 2.1, March 8, 1995.
- [GE88] Glasgow J., McEwen G. *Reasoning about knowledge and permission in secure distributed systems*. In Proc. of the computer security foundations workshop, Franconia, 1988.
- [GE90] Glasgow J., McEwen G. *A logic for reasoning about security*. In Proc. of the computer security foundations workshop, Franconia, 1990.
- [GGF98] Gligor V., Gavrilă S., Ferraiolo D., *On the Formal Definition of Separation-of -Duty Policies and their Composition* Proc. of the IEEE Symposium on Security and Privacy, 1998
- [GM84a] Goguen J., Meseguer J. *Security Policies and Security Models*, Symposium on Security and Privacy, pp.11-20,IEEE, April 1982.
- [GM84b] Goguen J., Meseguer J. *Unwinding, Inference Control*. In IEEE Symposium on Security, Privacy, Oakland, 1984.
- [God67] Gödel K., *On formally undecidable propositions of principia mathematica and related systems*. W: J. van Heijenoort (red.), From Frege to Godel. A Source Book in Mathematical Logic, pp. 596-661. Harward University Press, 1967.
- [Grz84] Grzegorzczak A., *Zarys logiki matematycznej.*, PWN, Warszawa 1984.
- [Gui95] Guiri L., *A new model for role-based access control*, In Proceedings of 11th Annual Computer Security Application Conference, pages 249-255, New Orleans, LA, December 11-15 1995.
- [Gum90] Gumański L., *Wprowadzenie w logikę współczesną.*, PWN, Warszawa 1990.
- [Ino91] Inoue K. *Consequencefinding based on ordered linear resolution* . In proc of IJCAI, 1991.

- [Hil81] Hilpinen R., *Deontic Logic: Introductory and Systematic Readings*, D.Reidel, Dordrecht, 1981.
- [ISO95] ISO/IEC 9075, Database Language SQL - Part 2: Foundation, Document ISO/IEC JTC1/SC21 N9463, March 1995.
- [ISO99] ISO/IEC 15408, *Common Criteria*, v.2.0, 1998.
- [JK99] Juszczyszyn K., Kołaczek G. *Analysis and Evaluation of Network Security Policies*, 2nd International Conference on Managing Enterprises, Newcastle, Australia. , November 1999.
- [JS93] Jones A.J.I. , Sergot M.J., *On the characterization of law and computer systems: the normative systems perspective*. In *Deontic Logic in Computer Science. Normative System Specification* pp. (275-307), Chichester, John Wiley & Sons, 1993
- [Kan72] Kanger S. *Law and Logic*. Theoria, 38, 1972.
- [Kol97] Kołaczek G., *The advantages of Distributed Management.*, XIX Int. Scientific School 'Information Systems Architecture and Technology', Szklarska Poręba, 1997.
- [Kol98] Kołaczek G., *Modelling Management Goals and Policies with Deontic Logic*, XX Int. Scientific School 'Information Systems Architecture and Technology', Szklarska Poręba, 1998.
- [Kol99a] Kołaczek G., *Analysis of Network Security Policy Consistency*, VIth Polish Teletraffic Symposium, Szklarska Poręba, April, 1999
- [Kol99b] Kołaczek G., *Analiza polityk bezpieczeństwa z wykorzystaniem logik modalnych*. Krajowe Sympozjum Telekomunikacji, cz. C, Bydgoszcz 1999.
- [Kol00a] Kołaczek G., *Zarządzanie dostępem do systemu informatycznego poprzez role*. Inżynieria Wiedzy i Syst. Ekspertowe IV, Wrocław, 2000.
- [Kol00b] Kołaczek G., *Effective Access Control*, XXII Int. Scientific School 'Information Systems Application and Technology', Szklarska Poręba, 2000.
- [Kri63] Kripke S.A., *Semantical analysis of modal logic I normal modal prepositional calculi*, Logic und Grundlagen d. Math Bd. 9. s.67-96, 1963
- [Kuh97] Kuhn R., *Mutual Exclusion of Roles as a Means of Implementing Separation of Duty in RBAC Systems*, Second ACM Workshop on RBAC, 1997
- [Kul99] Kulicki P., *Logika programowania, a sylogistyka Arystotelesa*, Rozprawa doktorska, KUL, Lublin 1999.
- [Lam74] Lampson B. *Protection*. In 5th Princeton Symposium on Information Sciences and Systems, March 1971. Reprinted in ACM Operating Systems Review, 8(1) (1974).
- [LHL74] Landwehr C., Heitmeyer C., McLean J., *A Security Model for military Message Systems*, March 1971, ACM Operating Systems Review 8 1974
- [Lea90] McLean J. *Security models and information flow*. In IEEE Symposium on Security and Privacy, Oakland 1990
- [Lea94] McLean J., *Security models*, In Encyclopedia of Software Engineering (ed. Marciniak J.), Wiley Press, 1994.

- [Lee67] Lee R. *A completeness theorem and computer program for finding theorems derivable from given axioms*. PhD thesis, Univ. of California, Berkeley, 1967.
- [Lee88] Lee T.M.P. *Using mandatory integrity to enforce commercial security*. In Proceedings IEEE Computer Society Symposium on Security and Privacy, pages 140-146, Oakland, CA, May 1988.
- [Loi94] Loiseau S. *Refinement of knowledge bases based on consistency*. In Proc of ECAI, 1994.
- [Loz91] Lozinski E.L., *Integrity in Automated Information Systems*. National Security Computer Center 1991
- [LS90] Lu, W., Sundaresan M., *A Model for Multilevel Security in Computer Networks*, IEEE Transactions on Software Engineering, Vol. 16, No. 6, Page 647, 1 June 1990.
- [Lun88] Lunt T. *Access control policies: Some unanswered questions*. In IEEE Computer Security Foundations Workshop II, pages 227-245, Franconia, NH, June 1988.
- [Mar87] Marciszewski W., *Logika formalna. Zarys encyklopedyczny z zastosowaniem do informatyki i lingwistyki*, PWN, Warszawa 1987.
- [NO95] Nyanchama M., Osborn S., *Access rights administration in rolebased security systems* . In J. Biskup, M. Morgernstern, and C. Landwehr, editors, Database Security VIII: Status and Prospects. NorthHolland, 1995.
- [MS91] Moffett J., Sloman M. *Delegation of Authority*. In Second IFIP TC6/WG6.6 International Symposium on Integrated Network Management, Washington DC, 1991.
- [MS93] Moffett J., Sloman M. *Policy hierarchies for distributed systems management*. IEEE JSAC, Special Issue on Network Management, 11(9), December 1993.
- [MS94] Moffett J., Sloman M. *Policy conflict analysis in distributed system management*. Journal of Organizational Computing, 1994.
- [NCSC85] National Computer Security Center, *Department of Defence Trusted Computer Security Evaluation Criteria*, DOD 5200.28-STD, 1985.
- [NCSC87] National Computer Security Center, *Trusted Network Interpretation*, NCSC-TG-005, 1987.
- [NCSC87a] National Computer Security Center, *A Guide to Understanding Discretionary Access Control In Trusted Systems*, raport NCSC-TG-003, September 30, 1987.
- [NCSC92] *A Guide to Understanding Security Modelling in Trusted Systems*, National Computer Security Center, Report NCSC-TG-010, 1992.
- [OL96] Ong K., Lee. R.M. *A decision support system for bureaucratic policy administration: abductive logic programming approach*., Decision Support Systems(16), 1996
- [Por77] Porn I. *Action Theory, Social Science; Some Formal Models*, volume 120 of Syntheses Library. D.Reidel, Dordrecht, 1977.
- [PoS70] Pogorzelski W.A., Słupecki J., *O dowodzie matematycznym*, PZWS, Warszawa 1970
- [PS94] Prakken H, Sergot M. *Contrary to Duty Imperatives, Defeasibility, Violability* . In A.J. I.Jones, M.Sergot, editors, Second International Workshop on Deontic Logic in Computer Science, Oslo, Norway, 1994.

- [PSu96] Porębska M., Suchoń W., *Elementarny wykład logiki formalnej z ćwiczeniami komputerowymi.*, Universitas, Kraków 1996.
- [RF98] Ramos P., Fiadeiro J.L., *A deontic logic for diagnosis of organisational process design*, in Proceedings of the Fourth International Workshop on Deontic Logic in Computer Science, pp. 353--369, (1998).
- [RL91] Ryu Y., Lee R. *Defeasible deontic reasoning : a logic programming model*. In Proc of DEON'91. Amsterdam, 1991.
- [San88] Sandhu R. *Transaction control expressions for separation of duties*. In Proceedings of 4th Aerospace Computer Security Conference, pages 282- 286, Orlando, FL, December 1988.
- [San90] Sandhu R. *Separation of duties in computerized information systems*. In Proceedings of the IFIP WG11.3 Workshop on Database Security, pages 18-21, Halifax, U.K., September 1990.
- [San92] Sandhu R. *Lattice-based enforcement of chinese walls*. Computers & Security, 11(8):753-763, December 1992.
- [San93] Sandhu R. *Lattice-based access control models*. IEEE Computer, 26(11):9-19, November 1993.
- [San96] Sandhu R. *Role hierarchies and constraints for lattice-based access controls*. In Elisa Bertino, editor, Proc. Fourth European Symposium on Research in Computer Security. Springer-Verlag, Rome, Italy, 1996. Published as Lecture Notes in Computer Science, Computer Security ESORICS96.
- [San97] Sandhu R. *Rationale for the RBAC96 family of access control models*. In Proceedings of the 1st ACM Workshop on Role-Based Access Control. ACM, 1997.
- [San98] Sandhu R., *Role activation hierarchies*. In Proceedings of 3rd ACM Workshop on Role-Based Access Control, Fairfax, VA, October 1998.
- [SB97] Sandhu R., Bhamidipati V. *The URA97 model for role-based administration of user-role assignment*. In T. Y. Lin and Xiaolei Qian, editors, Database Security XI: Status and Prospects. North-Holland, 1997.
- [SBCG97] Sandhu R, Bhamidipati V., Coyne E., Ganta S, Youman Ch.. *The ARBAC97 model for role-based administration of roles: preliminary description and outline*. In Proceedings of 2nd ACM Workshop on Role-Based Access Control, pages 41{50, Fairfax, VA., October 1997.
- [SBM99] Sandhu R, Bhamidipati V., Munawer Q. *The ARBAC97 model for role-based administration of roles*. ACM Transactions on Information and Systems Security, 2(1):105-135, February 1999.
- [SM98] Sandhu R. Munawer Q. *How to do discretionary access control using roles*. In Proceedings of 3rd ACM Workshop on Role-Based Access Control, pages 47-54, 1998.
- [SS97] Sandhu R., Samarati P., *Authentication, access control and intrusion detection*. In Elisa Bertino, editor, The Computer Science and Engineering Handbook, pages 1929{1948. CRC press, 1997.

- [SETA96] *A Marketing Survey of Civil Federal government Organizations to Determine the Need for Role Based Access Control Security Products* SETA Corporation 1996
- [SS94] Sandhu R. Samarati P., *Access Control: Principles and Practice*, IEEE Communications Magazine, 9/1994 pp.40-48.
- [SCFY94] Sandhu R.S., Coyne E. J., Feinstein H. L., Youman Ch. E.. *Role-based access control: A multi-dimensional view.* In Proceedings of 10th Annual Computer Security Application Conference, pages 54-62, December 1994.
- [SM94] Solms S. H., Merwe I., *The management of computer security profiles using a roleoriented approach* . Computers & Security, 8:673-680,1994.]
- [Suc80] Suchoń W., *Studia nad logiką deontyczną*, Rozprawy habilitacyjne nr 76 Uniwersytet Jagielloński, 1980
- [Sut86] Sutherland D. *A Model of Information.* In Proceedings of the 9th National Computer Security Conference, 1986.
- [Tho91] Thomsen D.J. *Rolebased application design and enforcement.* In S. Jajodia and C.E. Landwehr, editors, Database Security IV: Status and Prospects, pages 151-168. NorthHolland, 1991.
- [WM93] Wieringa R.J. , Meyer J.-J.Ch., *Applications of deontic logic to computer science: a concise overview.* In Deontic Logic in Computer Science. Normative System Specification pp. (17-40), Chichester, John Wiley & Sons, 1993
- [Zie68] Ziemba Z., *Paradoksy logiki deontycznej*, Państwo i Prawo z.1 s. 45-55 1968.
- [Zie69] Ziemba Z., *Logika deontyczna jako formalizacja rozumowań normatywnych*, Warszawa 1969.

