Sovietique «Informatika-91». – Grenoble, France. – 16–19 October, 1991. – P. 99–111.
9. Gentzen G. Untersuchungen uber das Logische Schliessen / G. Gentzen. – Math. Z. – 1934. – Vol. 39. – P. 176–210.
10. Lyaletski A. Goal-driven inference search in classical propositional logic / A. Lyaletski, A. Paskevich. – Proc. International Workshop STRATEGIES'2001, Siena, Italy. – 2001. – P. 65–74.
11. Минц Г. Е. Теорема Эбрана / Г. Е. Минц // Математическая теория логического вывода. – М. : Наука, 1967. – С. 311–350.

A. Afonin

ON COMPUTER-ORIENTED SEQUENT-TYPE CALCULI FOR FIRST-ORDER CLASSICAL LOGIC

The paper is devoted to an approach to the construction of computer-oriented sequent-type calculi that is not required fulfilling the preliminary skolemization operation. Its peculiarity is demonstrated with the help of the construction of a special calculus being investigated on the soundness and completeness. Using the calculus, methods for complete extensions of the SLD-resolution both for sets of arbitrary clauses and for sets of usual formulas of the 1st order language are described.

УДК 518.74

Коляденко А. А.

ВИКОРИСТАННЯ ЗАСОБІВ АВТОМАТИЧНОГО ДОВЕДЕННЯ ТЕОРЕМ ДЛЯ ДОСЛІДЖЕННЯ МОДЕЛЕЙ КОНТРОЛЮ ДОСТУПУ НА БАЗІ РОЛЕЙ ТА ГЕНЕРАЦІЇ АВТОРИЗАЦІЙНИХ ТВЕРДЖЕНЬ

Розглянуто проблематику застосування засобів автоматичного доведення теорем для дослідження моделей контролю доступу на базі ролей. Запропоновано шляхи представлення моделей стандарту ANSI-INCITS 359-2004 Role Based Access Control як теорій першого порядку, дослідження їх властивостей та генерації авторизаційних рішень за допомогою ПЗ EProver.

1. Вступ

Одним з найпопулярніших підходів в цій галузі є контроль доступу на базі ролей (Role Based Access Control, або RBAC). Головна риса RBAC зводиться до того, що користувач не отримує прямого доступу до ресурсів інформаційної системи. Натомість такий доступ реалізується опосередковано через поняття ролі. За такого підходу для надання доступу користувачеві до певного ресурсу інформаційної системи встановлюється відповідність користувача деякій ролі, яка, своєю чергою, отримує доступ безпосередньо до ресурсу. На думку автора, такий шлях має дві переваги:

- система контролю доступу стає більш гнучкою (користувач може бути від'єднаним або приєднаним до ролі без зміни її властивостей або під час зміни властивостей ролі. Ця зміна автоматично впливає на всіх користувачів, що є чи будуть приєднаними до ролі) порівняно, наприклад, з традиційними Access Control Lists та Discretionary Access Control [5];
- поняття ролей дає змогу вдало відображати структуру організації, яка експлуатує цільову інформаційну систему, на об'єкти механізмів контролю доступу (наприклад, посаду account manager можна авторизувати роллю RBAC Account Manager).

Завдяки цим перевагам контроль доступу на основі ролей реалізовано в багатьох провідних проектах програмного забезпечення (операційні

системи, системи керування базами даних тощо). Беручи до уваги ці переваги, автор вважає RBAC найкращим прикладом моделі контролю доступу для розв'язанні задачі контролю доступу в розподіленій інформаційній системі з складною організаційною структурою.

Єдиним стандартом RBAC сьогодні є *ANSI-INCITS 359-2004 Role Based Access Control* [1]. Цей стандарт описує кілька моделей контролю доступу на базі ролей. Наразі в роботі розглядаються моделі Core RBAC та Hierarchical RBAC цього стандарту.

Головна мета роботи полягає у формулюванні моделей Core RBAC та Hierarchical RBAC як аксіоматичних теорій першого порядку та формального дослідження деяких властивостей цих моделей. Запропоновано підхід щодо використання засобів автоматичного доведення теорем для дослідження властивостей моделей контролю доступу та генерації авторизаційних тверджень.

2. Role Based Access Control

За всю історію використання RBAC як засобу контролю доступу до ресурсів було створено багато реалізацій RBAC, які відрізняються одна від одної. У цьому розділі розглянуто найбільш важливі та спільні для всіх реалізацій риси RBAC.

Будь-яка інфраструктура контролю доступу базується на деякій функції авторизації auth(), яка відповідає на питання, чи дозволити користувачеві виконання певної операції над деяким ресурсом. Конфігуруючи інфраструктуру контролю доступу, задача адміністратора інформаційної системи – визначити функції авторизації. Мета будь-яких способів організації контролю доступу зводиться до автоматизації процесу задання функції авторизації, забезпечення гнучкості та масштабованості цього процесу.

Припустимо, в контексті деякої інформаційної системи визначені множини користувачів U, операцій O та ресурсів Re. Тоді в найпростішому випадку функцію авторизації можна задати множиною четвірок $\{(u, o, re, auth\_result) \mid u \in U, o \in O, re \in Re, auth\_result \in \{true, false\}\}$. На відміну від такого способу задання функції авторизації, що характеризується визначенням прямих дозволів користувачам, RBAC вводить проміжну ланку між користувачами та ресурсами – ланку ролей. У випадку RBAC, для дозволу певному користувачеві виконання деякої операції над ресурсом, його необхідно авторизувати роллю, що має дозвіл на виконання такої операції. Такий підхід надає додаткову гнучкість процесу конфігурації інфраструктури контролю доступу: за допомогою однієї зміни у інфраструктурі, користувачеві може бути наданий доступ до групи дозволів, уособлених роллю; за допомогою однієї зміни у інфраструктурі, дозвіл може бути поширений на групу користувачів, що відповідають певній ролі.

На перший погляд, поняття ролі дуже схоже на поняття групи користувачів. Фактично, в найпростіших моделях ці поняття ідентичні, різниця лише у метафорі: роль – це набір дозволів, натомість група – це набір користувачів. У розширенні RBAC ця різниця у метафорах є суттєвою, що не дає можливості ототожнювати поняття ролі та групи користувачів у моделях RBAC, наприклад, з ієрархією ролей (будуть розглянуті нижче).

Оскільки мета цього розділу не полягає в формулюванні певної формальної моделі RBAC (це буде зроблено нижче), проілюструємо такий підхід на прикладі.

Приклад 1. Припустимо, деяка компанія BigCompany, котра спеціалізується на виробництві та продажу комп'ютерної техніки, для роздрібного продажу своєї продукції використовує web-орієнтовану інформаційну систему InetStore, в якій для контролю доступу до ресурсів використовується RBAC. В департаменті роздрібного продажу з системою InetStore працюють: головний оператор продажу John та молодший оператор продажу Richard. В процесі функціонування кожен з працівників має такі повноваження:

- головний оператор продажу John може створювати та видаляти розділи. Також він може створювати, редагувати та видаляти з продажу лоти певного товару (сервер, ноутбук тощо). Кожен з лотів може бути доданий до будь-якого (лише одного) раніше створеного розділу;
- молодший оператор продажу Richard може створювати, редагувати та видаляти лоти з продажу товару.

Припустимо, зараз в системі InetStore створено розділ Laptops, в якому містяться лоти: A, створений John-ом, та B, створений Richard-ом. Тоді поточний стан RBAC можна зобразити як на рис. 1.

Як бачимо, в системі створено дві ролі: Folder Admin та Lot Admin. Роль Folder Admin має дозволи на редагування та видалення розділу Laptops, а роль Lot Admin має дозволи на редагування і видалення лотів A та C.

Слід зауважити, що John, Richard, Laptops, A і B є об'єктами контролю доступу на базі ролей. Зазвичай (хоча і не обов'язково) вони бієктивно відображають об'єкти інформаційної системи на об'єкти підсистеми контрою доступу.

У прикладі був вказаний один з варіантів використання моделі Core RBAC стандарту *ANSI-INCITS 359-2004 Role Based Access Control*, яка

формалізує найпростіші засоби контролю доступу на базі ролей.

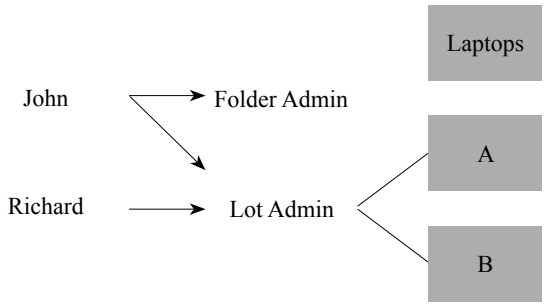

Рис. 1. Поточний стан RBAC

В наступних розділах будуть зображені формальні моделі управління доступом на базі ролей. Будемо вважати, що для всіх зазначених нижче випадків будуть справедливі такі загальні припущення:

– інфраструктура контролю доступу в певний момент часу перебуває в певному стані, який характеризується вектором множин. Цей вектор ми будемо називати станом інфраструктури контролю доступу;
– для виконання певної операції над деяким ресурсом користувач надсилає в інформаційну систему запит, який містить посилання на операцію, ресурс та іншу інформацію;
– запит перехоплює інфраструктура контролю доступу, яка приймає рішення про дозвіл або не дозвіл виконання операції, базуючись на обчисленому значенні функції авторизації;
– в обчисленні значення функції авторизації використовується інформація з запиту та поточний стан інфраструктури контролю доступу.

3. ANSI-INCITS 359-2004 Role Based Access Control

Стандарт *ANSI-INCITS 359-2004 Role Based Access Control* був розроблений на основі моделей, викладених в роботах [2–4]. У стандарті викладено три моделі: Core RBAC, Hierarchical RBAC та Constrained RBAC. Кожна з моделей характеризується двома об'єктами: структурами даних, що задають політику контролю доступу на основі ролей, і набором функцій, які призначені для модифікації цієї політики та прийняття авторизаційного рішення. У цьому розділі буде побудовано математичну модель стандарту та розглянуто деякі властивості цієї моделі.

3.1. ANSI-INCITS 359-2004 Core RBAC

Модель Core RBAC задає базову функціональність контролю доступу на основі ролей, яка засновується на чотирьох множинах:

U – користувачі,

Ro – ролі,

O – операції,

Re – ресурси.

Модель також базується на двох відношеннях:

1. $URo \subset U \times Ro$ – відповідність користувачів ролям. Так, (u, ro) ∈ URo відображає відповідність користувача u ролі ro.

2. RoORe $\subset Ro \times O \times Re$ – дозволи ролям виконувати операції над ресурсами. Так, (ro, o, re) ∈ RoORe відображає відповідність користувача u ролі ro.

Позначимо $(u, ro) \in URo$, як $u \xrightarrow{URO} ro$ – користувач u відповідає ролі ro.

Позначимо $(ro, o, re) \in RoORe$, як $ro\ (o) \xrightarrow{ROORE} re$ – ролі ro дозволено виконувати операції o над ресурсом re.

Назвемо шістку (U, Ro, O, Re, URo, RoORe) – станом Core RBAC. У певний момент часу інформаційна система з контролем доступу на основі моделі Core RBAC може мати лише один стан Core RBAC. Стан, в якому інфраструктура контролю доступу перебуває в поточний момент, назвемо поточним станом Core RBAC. Іншими словами, інформаційну систему можна представити як автомат, а поточний стан Core RBAC буде складовою структури даних, що відображає стан цього автомату в певний час. (c1)

Під час спроби користувача u провести операцію o над ресурсом re, інформаційна система приймає рішення, чи дозволити користувачу провести цю операцію (авторизаційне рішення). Рішення приймається базуючись на обчисленому значенні функції авторизації auth_func (u, o, re), де auth_func для деякого поточного стану Core RBAC = (U, Ro, O, Re, RoORe, RoORe) визначається:

auth_func: $U \times O \times Re \rightarrow$ {true, false},

auth_func(u:NAME, o:NAME, re:NAME) :

return $\exists$ ro ∈ Ro, така що $u \rightarrow ro \wedge ro\ (o) \xrightarrow{ROORE} re$.

Для $\forall$ u ∈ U, o ∈ O, re ∈ Re будемо позначати $u\ (o) \xrightarrow{UORE} re$, якщо auth_func(u, o, re) = true.

3.2. Представлення моделі ANSI-INCITS 359-2004 Core RBAC у вигляді теорії першого порядку

Тут модель ANSI-INCITS 359-2004 Core RBAC буде формалізовано як теорію першого порядку. Слід зазначити, що певному стану Core RBAC відповідатиме одна і тільки одна модель сформульованої нижче теорії.

Визначимо мову першого порядку як трійку

$$\text{CoreRBACLang} = (Cn^C, Fn^C, \text{Pr}^C),$$

де $Cn^C = \{cn_1, cn_2, \ldots\}$ – множина констант, що можуть залежно від інтерпретації позначати ролі, користувачів, операції та ресурси,

Fn^C – пуста множина функціональних символів,

Pr^C = {U, O, Re, Ro, $() \xrightarrow{ROORE}$, $() \xrightarrow{UORE}$, $\xrightarrow{URO}$} – множина предикатних символів, при цьому предикатні символи U, O, Re, Ro є унарними предикатними символами, $\xrightarrow{URO}$ – бінарні предикатні символи, а $() \xrightarrow{ROORE}$ та $() \xrightarrow{UORE}$ – предикатні символи арності 3.

На базі мови першого порядку CoreRBACLang побудуємо теорію першого порядку CoreRBACTheory, спираючись на множину аксіом:

- $A \to (A \to B)$; (a1)
- $(A \to B) \to ((A \to (B \to C)) \to (A \to C))$; (a2)
- $A \wedge B \to A$; (a3)
- $A \wedge B \to B$; (a4)
- $A \to (B \to A \wedge B)$; (a5)
- $A \to A \vee B$; (a6)
- $B \to A \vee B$; (a7)
- $(A \to C) \to ((B \to C) \to (A \vee B \to C))$; (a8)
- $(A \to B) \to ((A \to \neg B) \to \neg A)$; (a9)
- $\neg \neg A \to A$; (a10)
- $\forall v\, A(v) \to A(t)$; (a11)
- $A(t) \to \exists\, v\, A(v)$; (a12)
- $\forall cn, (\neg (O(cn) \vee Re(cn) \vee Ro(cn)) \vee \neg (U(cn) \vee Re(cn) \vee Ro(cn)) \vee \neg (O(cn) \vee U(cn) \vee Ro(cn)) \vee \neg (O(cn) \vee Re(cn) \vee U(cn))) \wedge (O(cn) \vee Re(cn) \vee Ro(cn) \vee U(cn))$, (1т)
- $\forall u,o,re, ((U(u) \wedge O(o) \wedge Re(re) \wedge u\,(o) \xrightarrow{UORE} re) \leftrightarrow (U(u) \wedge O(o) \wedge Re(re) \wedge \exists ro,\, u \xrightarrow{URO} ro \wedge ro\,(o) \xrightarrow{ROORE} re))$, (2т)
- $\forall u,ro\ (U(u) \wedge Ro(ro) \wedge u \xrightarrow{URO} ro) \leftrightarrow (U(u) \wedge Ro(ro) \wedge \forall o,re\ (O(o) \wedge Re(re) \wedge ro\ (o) \xrightarrow{ROORE} re) \to (O(o) \wedge Re(re) \wedge u\ (o) \xrightarrow{UORE} re))$, (3т)
- $\forall\ ro,o,re, (Ro(ro) \wedge O(o) \wedge Re(re) \wedge ro\ (o) \xrightarrow{ROORE} re) \leftrightarrow (Ro(ro) \wedge O(o) \wedge Re(re) \wedge \forall u\ (U(u) \wedge u \xrightarrow{URO} ro) \to (U(u) \wedge u\ (o) \xrightarrow{UORE} re))$. (4т)

Всі моделі теорії CoreRBACTheory мають такі властивості:

- кожна модель відповідає певному стану Core RBAC = ($U^{CoreRBAC}$, $Ro^{CoreRBAC}$, $O^{CoreRBAC}$, $Re^{CoreRBAC}$, $URo^{CoreRBAC}$, $RoO Re^{CoreRBAC}$);
- множина констант Cn^C визначається як $U^{CoreRBAC} \bigcup Ro^{CoreRBAC} \bigcup O^{CoreRBAC} \bigcup Re^{CoreRBAC} \bigcup$ VARS, де VARS – множина символів змінних;
- для певного cn $\in Cn^C$ предикат U(cn) визначається так: якщо cn $\in U^{CoreRBAC}$, то результат предикату дорівнює true, у іншому випадку результат предикату дорівнює false. Оскільки в теорії першого порядку CoreRBACTheory немає неатомарних термів (через порожність Fn), то немає потреби поширювати дію предикату на множину термів мови першого порядку CoreRBACLang;
- для певного cn $\in Cn^C$ предикат Ro(cn) визначається таким чином: якщо cn $\in Ro^{CoreRBAC}$, то результат предикату дорівнює true, у іншому випадку результат предикату дорівнює false. Оскільки в теорії першого порядку CoreRBACTheory немає неатомарних термів (через порожність *Fn*), то немає необхідності поширювати дію предикату на множину термів мови першого порядку CoreRBACLang;
- для певного cn $\in Cn^C$ предикат O(cn) визначається так: якщо cn $\in O^{CoreRBAC}$, то результат предикату дорівнює true, у іншому випадку результат предикату дорівнює false. Оскільки в теорії першого порядку CoreRBACTheory немає неатомарних термів (через порожність Fn), то немає необхідності поширювати дію предикату на множину термів мови першого порядку CoreRBACLang;
- для певного cn $\in Cn^C$ предикат Re(cn) визначається так: якщо cn $\in Re^{CoreRBAC}$, то результат предикату дорівнює true, у іншому випадку результат предикату дорівнює false. Оскільки в теорії першого порядку CoreRBACTheory немає неатомарних термів (через порожність Fn), то немає потреби поширювати дію предикату на множину термів мови першого порядку CoreRBACLang;
- предикат $() \xrightarrow{ROORE}$ (x, y, z) (далі будемо позначати як x (y) $\xrightarrow{ROORE}$ z) визначимо як true, якщо Ro(x) $\wedge$ O(y) $\wedge$ Re(z) $\wedge \exists$ (x, y, z) $\in RoO Re^{CoreRBAC}$ (x є роллю, y – операцією та z – ресурсом), та false в іншому випадку;
- предикат $\xrightarrow{URO}$ (x, y) (далі позначатимемо як x $\xrightarrow{URO}$ y) визначимо як true, якщо U(x) $\wedge$ Ro(y) $\wedge \exists$ (x, y) $\in URo^{CoreRBAC}$;
- предикат $() \xrightarrow{UORE}$ (x, y, z) (далі позначатимемо x (y) $\xrightarrow{UORE}$ z), визначимо еквівалентним формулі: $\exists$ r, x $\xrightarrow{URO}$ r $\wedge$ r (y) $\xrightarrow{ROORE}$ z .

Аксіоми (a1) – (a12) є аксіомами класичного числення предикатів.

Аксіома (1т) визначає щодо моделей теорії вимогу належності певного елемента з Cn лише до одного класу із зазначених: користувачі, ролі, операції та ресурси.

Аксіома (2т) визначає відносно моделей теорії вимогу визначення предиката x (y) $\xrightarrow{ROORE}$ z у вигляді формули $\exists$ ro, Ro(ro) $\wedge$ u $\xrightarrow{URO}$ ro $\wedge$ ro (o) $\xrightarrow{ROORE}$ re.

Аксіоми (3т) та (4т) визначають додаткові очевидні співвідношення між предикатами $() \xrightarrow{ROORE}$, $() \xrightarrow{UORE}$ та $\xrightarrow{URO}$, що розширюють можливості теорії у виведенні теорем.

3.3. ANSI-INCITS 359-2004 Hierarchical RBAC

Модель Hierarchical RBAC відрізняється від моделі Core RBAC ієрархією ролей. Типове використання цієї моделі полягає в проектуванні ієрархічної структури працівників організації на

ієрархію ролей, наприклад, посаді начальника відділу відповідатиме роль начальника відділу. Докладніше такий підхід буде продемонстровано нижче.

Формально ієрархія ролей задається відношенням $RoH \subset Ro \times Ro$.

В моделі Hierarchical RBAC поняття стану визначимо як сімку (U, Ro, O, Re, RoORe, RoORe, RoH), де RoH – відношення, що задає ієрархію ролей, а решта множин мають таке саме значення, що і в Core RBAC.

Позначимо $(ro_1, ro_2) \in RoH$, як $ro_1 < ro_2$ – роль ro_2 є батьківською щодо ro_1, що, своєю чергою, означає, що всі дозволи, встановлені для ro_2, поширюються також на ro_1.

Формально це виражається так: $ro_1 < ro_2 \Rightarrow (\forall o, re, (ro_2\ (o) \xrightarrow{ROORE} re \Rightarrow ro_1\ (o) \xrightarrow{ROORE} re))$. (1b)

У стандарті *ANSI-INCITS 359-2004 Role Based Access Control* існує також інше визначення відношення RoH: якщо для деяких ro_1 та ro_2 існує $(ro_1, ro_2) \in RoH$, то це позначається як $ro_1 < ro_2$ – роль ro_2 є батьківською щодо ro_1, тобто всі користувачі, які авторизовані стосовно ролі ro_1, є авторизованими і до ro_2.

Формально це виражається так: $ro_1 < ro_2 \Rightarrow (\forall u, (u \xrightarrow{URO} ro_1 \Rightarrow u \xrightarrow{URO} ro_2))$. (1c)

Насправді ці два означення є еквівалентними, тобто:

$(\forall o, re, (ro_2\ (o) \xrightarrow{ROORE} re \Rightarrow ro_1\ (o) \xrightarrow{ROORE} re)) \Leftrightarrow (\forall u, (u \xrightarrow{URO} ro_1 \Rightarrow u \xrightarrow{URO} ro_2))$. (1e)

Ми доведемо цей факт пізніше, після того, як сформулюємо модель Hierarchical RBAC як теорію першого порядку.

Позначимо $ro_1\ (\ldots) < ro_n$, якщо існують такі $ro_2, ..., ro_{n-1}$, для яких виконується умова: $\forall 1 \le i < n$, $ro_i < ro_{i+1}$. Також будемо вважати: для $\forall$ ro справедливо, що ro (…)< ro.

Для деякого користувача u, якщо існують такі ro, ro_1, що $u \rightarrow ro$ та $ro\ (...) < ro_1$, то позначимо цей факт як $u\ (\ldots) \rightarrow ro_1$.

Також введемо предикат $()(...) \xrightarrow{ROORE}$: позначимо ro $(o)(...) \xrightarrow{ROORE}$ re, якщо $\exists\ ro_1$, така, що ro $(...) < ro_1$, та $ro_1\ (o) \xrightarrow{ROORE}$ re. Визначимо як u $(o)(\ldots) \xrightarrow{UORE}$ re, якщо $\exists$ ro $\in$ Ro, ro $(o)(...) \xrightarrow{ROORE}$ re.

Отже, крім предикатів $() \xrightarrow{ROORE}$, $() \xrightarrow{UORE}$, $\xrightarrow{URO}$, та <, що задаються адміністратором системи контролю доступу за допомогою визначення множин U, Ro, O, Re, RoORe, RoORe та RoH, ми визначили також предикат непрямого наслідування (...)<, який отримується за допомогою транзитивного замикання відношення RoH, а також предикати $()(...) \xrightarrow{ROORE}$, $(...) \xrightarrow{UORE}$, $(...) \xrightarrow{URO}$, котрі визначаються ієрархією ролей RoH.

3.4. Представлення моделі ANSI-INCITS 359-2004 Hierarchical RBAC у вигляді теорії першого порядку

Розширимо теорію першого порядку CoreRBACTheory новими аксіомами, які дадуть нам змогу задавати стани ANSI-INCITS 359-2004 Hierarchical RBAC у вигляді інтерпретацій нової теорії. Назвемо розширену теорію HierarchicalRBACTheory.

Визначимо мову першого порядку HierarchicalRBACLang, задавши її як тріку HierarchicalRBACLang = $(Cn^H, Fn^H, \Pr^H)$, де $Cn^H = Cn^C$, $Fn^H = Fn^C$, $\Pr^H = \Pr^C \cup \{<, (\ldots)<, ()(...) \xrightarrow{ROORE}, ()(...) \xrightarrow{UORE}\}$.

Для побудови теорії першого порядку HierarchicalRBACTheory на базі мови першого порядку HierarchicalRBACLang нам потрібно ввести одну аксіому, яка б базувалася на співвідношенні (1b) або (1c). Покажемо, що ці співвідношення є еквівалентними в теорії CoreRBACTheory.

Теорема 1. Формула

$(\forall o, re, (ro_2\ (o) \xrightarrow{ROORE} re \Rightarrow ro_1\ (o) \xrightarrow{ROORE} re)) \Leftrightarrow (\forall u, (u \xrightarrow{URO} ro_1 \Rightarrow u \xrightarrow{URO} ro_2))$ (т3)

є всюди дійсною в теорії CoreRBACTheory.

Доведення цієї теореми не є тривіальним і було зроблено за допомогою автоматичних засобів доведення теорем, тому воно буде розглянуто в наступних розділах.

Базуючись на співвідношенні (1b), введемо наступну аксіому, яка б визначала предикат, що задає ієрархію ролей:

$\forall\ ro_a, ro_b\ ((\mathrm{Ro}(ro_a) \wedge \mathrm{Ro}(ro_b) \wedge ro_a < ro_b) \leftrightarrow \forall\ o,re(\mathrm{Ro}(ro_a) \wedge \mathrm{Ro}(ro_b) \wedge (\mathrm{O}(o) \wedge \mathrm{Re}(re) \wedge ro_b\ (o) \xrightarrow{ROORE} re) \rightarrow (\mathrm{O}(o) \wedge \mathrm{Re}(re) \wedge ro_a\ (o) \xrightarrow{ROORE} re)))$, (13т)

аксіому, що встановлювала б взаємозв'язок між предикатами < та (…)<:

$\forall\ ro_a, ro_b\ (\mathrm{Ro}(ro_a) \wedge \mathrm{Ro}(ro_b) \wedge ro_a\ (\ldots) < ro_b) \leftrightarrow (\mathrm{Ro}(ro_a) \wedge \mathrm{Ro}(ro_b) \wedge (ro_a < ro_b \vee \exists\ ro_c, \mathrm{Ro}(\mathrm{roI} \wedge ro_a\ (\ldots) < ro_c \wedge ro_c\ (\ldots) < ro_b)))$, (14т)

аксіому рефлективності відношення (…)<:

$\forall$ ro, Ro(ro) $\rightarrow$ (Ro(ro) (…)< Ro(ro)), (15т)

аксіоми, що визначали б предикати $()(...) \xrightarrow{ROORE}$ та $()(...) \xrightarrow{UORE}$:

$\forall$ ro,o,re (Ro(ro) $\wedge$ O(o) $\wedge$ Re(re) $\wedge$ ro (o)(...) $\xrightarrow{ROORE}$ re) $\leftrightarrow \exists\ ro_a$ (Ro(ro) $\wedge$ O(o) $\wedge$ Re(re) $\wedge$ $\mathrm{Ro}(ro_a) \wedge$ ro (…)$< ro_a \wedge ro_a$ (o) $\xrightarrow{ROORE}$ re), (16т)

$\forall$ u,o,re (U(u) $\wedge$ O(o) $\wedge$ Re(re) $\wedge$ u (o)(...) $\xrightarrow{UORE}$ re) $\leftrightarrow \exists$ro (U(u) $\wedge$ O(o) $\wedge$ Re(re) $\wedge$ Ro(ro) $\wedge$ u $\xrightarrow{URO}$ ro $\wedge$ ro (o)(...) $\xrightarrow{ROORE}$ re). (17т)

4. Дослідження властивостей моделей контролю доступу за допомогою засобів автоматичного доведення теорем

Вище ми сформулювали дві моделі контролю доступу стандарту *ANSI-INCITS 359-2004 Role*

Based Access Control, що дає нам можливість використовувати засоби автоматичного доведення теорем для розв'язання таких типів задач:

– дослідження властивостей моделей: визначити, чи є певні предикати всюди істинними в деякій моделі, перевірити певні співвідношення між декількома моделями;
– генерація авторизаційних рішень – використання засобів автоматичного доведення теорем як модуля контролю доступу інформаційної системи.

У наступних розділах розглянуто приклади розв'язання цих типів задач.

У прикладах, що будуть наведені, використано програмне забезпечення EProver [7], що є засобом автоматичного доведення теорем. У випадку цього ПЗ проблема записується як набір аксіом і теорем, що повинні бути доведені. ПЗ EProver може обробляти кілька мов завдання проблем. Всі розглянуті нижче приклади записані мовою TSTP [8], що також підтримується ПЗ EProver.

У створенні нових або використанні існуючих моделей контролю доступу (наприклад, на базі ролей) досліднику може знадобитися відповісти на запитання: чи має певну властивість модель або група моделей контролю доступу? Один із формальних підходів дослідження цього питання є формування моделі контролю доступу у вигляді теорії першого порядку, як це було показано в попередніх розділах, та формулювання властивості у вигляді формули цієї теорії. І тому відповідь на досліджуване запитання перетворюється у відповідь на інше: чи виводиться формула властивості в теорії першого порядку моделі контролю доступу? Пошук шляху виводу формули може виявитися досить складною задачею. Для автоматизації пошуку виведення формули властивості у теорії першого порядку моделі контролю доступу в роботі пропонується використовувати засоби автоматичного доведення теорем.

Розглянемо задачу доведення теореми 1. Нам потрібно довести, що формула (т3) є всюди дійсною в теорії CoreRBACTheory, для цього достатньо довести, що формула (т3) виводиться в теорії CoreRBACTheory. Покажемо, як можна це перевірити за допомогою ПЗ автоматичного доведення теорем EProver.

Як зазначалося, ми повинні дати на вхід ПЗ EProver файл проблеми, що містить множину аксіом і теорему, задані мовою TSTP. Відповідно до задачі, що розглядається як аксіома, у нас будуть формули (1т), (2т), (3т) і (4т) (формули (1а) – (12а) вже містяться у ПЗ EProver у якості аксіом), а теорема, яку треба довести, задається формулою (т3).

Тобто множина аксіом складається з таких формул теорії:

- $\forall$cn, $\neg$(O(cn) $\vee$ Re(cn) $\vee$ Ro(cn)) $\vee$ $\neg$(U(cn) $\vee$ Re(cn) $\vee$ Ro(cn)) $\vee$ $\neg$ (O(cn) $\vee$ U(cn) $\vee$ Ro(cn)) $\vee$ $\neg$ (O(cn) $\vee$ Re(cn) $\vee$ U(cn)) $\wedge$ (O(cn) $\vee$ Re(cn) $\vee$ Ro(cn) $\vee$ U(cn)), (1т)

- $\forall$u,o,re, ((U(u) $\wedge$ O(o) $\wedge$ Re(re) $\wedge$ u (o) $\xrightarrow{UORE}$ re) $\leftrightarrow$ (U(u) $\wedge$ O(o) $\wedge$ Re(re) $\wedge$ $\exists$ro, u $\xrightarrow{URO}$ ro $\wedge$ ro (o) $\xrightarrow{ROORE}$ re)), (2т)

- $\forall$u,ro (U(u) $\wedge$ Ro(ro) $\wedge$ u $\xrightarrow{URO}$ ro) $\leftrightarrow$ (U(u) $\wedge$ Ro(ro) $\wedge$ $\forall$o,re (O(o) $\wedge$ Re(re) $\wedge$ ro (o) $\xrightarrow{ROORE}$ re) $\rightarrow$ (O(o) $\wedge$ Re(re) $\wedge$ u (o) $\xrightarrow{UORE}$ re)), (3т)

- $\forall$ ro,o,re, (Ro(ro) $\wedge$ O(o) $\wedge$ Re(re) $\wedge$ ro (o) $\xrightarrow{ROORE}$ re) $\leftrightarrow$ (Ro(ro) $\wedge$ O(o) $\wedge$ Re(re) $\wedge$ $\forall$u (U(u) $\wedge$ u $\xrightarrow{URO}$ ro) $\rightarrow$ (U(u) $\wedge$ u (o) $\xrightarrow{UORE}$ re)), (4т)

а теорема, яку треба доведести, задається формулою, що нижче:

$(\forall o, re, (ro_2(o) \xrightarrow{ROORE} re \Rightarrow ro_1(o) \xrightarrow{ROORE} re)) \Leftrightarrow (\forall u, (u \xrightarrow{URO} ro_1 \Rightarrow u \xrightarrow{URO} ro_2))$. (т3)

Зазначена задача може бути описана мовою TSTP так:

```
% axiom #1
fof(a1, axiom, ![C]:(~(o(C) | re(C) | ro(C)) | ~(u(C) | re(C) | ro(C)) | ~(o(C) | u(C) | ro(C)) | ~(o(C) | re(C) | u(C)) & (o(C) | re(C) | ro(C) | u(C)))).
% axiom #2
fof(a2, axiom, ![X,Y,Z] : ((u(X) & o(Y) & re(Z) & uore(X,Y,Z)) <=> ? [RO] : (u(X) & o(Y) & re(Z) & ro(RO) & uro(X,RO) & roore(RO,Y,Z)))).
% axiom #3
fof(a3, axiom,![U,RO]:((u(U) & ro(RO) & uro(U,RO))<=>(u(U) & ro(RO) &! [O,RE] :((o(O) & re(RE) & roore(RO,O,RE))=>(o(O) & re(RE) & uore(U,O,RE))))))).
% axiom #4
fof(a4, axiom, ! [RO,O,RE]:((ro(RO) & o(O) & re(RE) & roore(RO,O,RE))<=>(ro(RO) & o(O) & re(RE) &! [U] :((u(U) & uro(U,RO))=>(u(U) & uore(U,O,RE))))))).
% problem formula
fof(c1, conjecture,! [RO1,RO2] :((ro(RO1) & ro(RO2) &! [O,RE] :((o(O) & re(RE) & roore(RO2,O,RE))=>( o(O) & re(RE) & roore(RO1,O,RE)))) <=> (ro(RO1) & ro(RO2) & ! [U] : ((u(U) & uro(U,RO1)) => (u(U) & uro(U,RO2))))))).
```

На жаль, ПЗ EProver не має опції виведення шляху доведення формули у вигляді, що може бути зрозумілим людині. Натомість ПЗ EProver, залежно від набору вхідних опцій, виводить або лише кінцевий результат (чи доведено формулу, чи ні), або всі проміжні обчислення. Тому автор розробив програму TSTPExtractor. Програма написана мовою Java і призначена для перетворення важкозрозумілого вихідного файлу ПЗ EProver у вигляд, який може зрозуміти людина. Увесь процес доведення формули показаний на рис. 2.

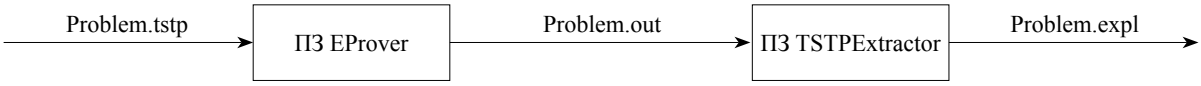

Рис. 2. Шлях доведення формули

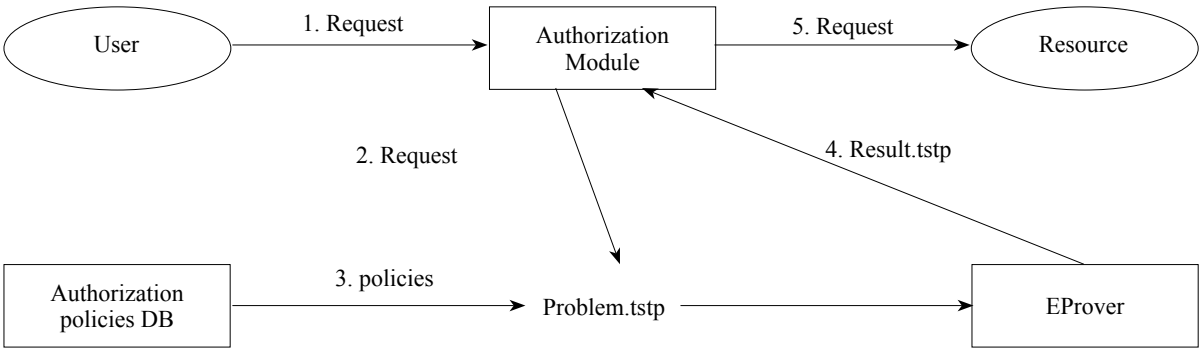

Рис. 3. Принцип роботи модуля

Розглянемо деякі приклади входжень файлу Problem.expl. Весь файл не зазначено, оскільки він містить більше ста виводів.

=> c_0_5 (![X9] : ![X10] : (((ro(X9) & ro(X10)) & ![X7] : ![X8] : (((o (X7) & re(X8)) & roore(X10,X7,X8)) => ((o(X7) & re(X8)) & roore(X9,X7,X8)))) <=> ((ro(X9) & ro(X10)) & ![X6] : ((u(X6) & uro(X6,X9)) => (u(X6) & uro(X6,X10))))))).

Тут вираз «=> c_0_5 formula» вказує на те, що formula є аксіомою з іменем c_0_5. З постановки задачі зрозуміло: формулу насправді потрібно довести. Тут криється важливий принцип роботи ПЗ EProver (і багатьох інших засобів автоматичного доведення теорем): ПЗ EProver до набору аксіом додає аксіому, що заперечує формулу, яку потрібно довести, а потім з отриманої множини аксіом намагається вивести формулу false. Якщо виведення знайдено, то, відповідно, виводиться й формула проблеми (за принципом від супротивного). Наступний рядок демонструє отримання з аксіоми c_0_5 нової формули c_0_6, що є запереченням аксіоми c_0_5:

c_0_5 => c_0_6 (~ (![X9] : ![X10] : (((ro(X9) & ro(X10)) & ![X7] : ![X8] : (((o(X7) & re(X8)) & roore(X10,X7,X8)) => ((o(X7) & re(X8)) & roore(X9,X7,X8)))) <=> ((ro(X9) & ro(X10)) & ![X6] : ((u(X6) & uro(X6,X9)) => (u(X6) & uro(X6,X10)))))))).

Останній рядок файлу Problem.expl: c_0_525 => c_0_526 ($false) – означає, що з множини аксіом, яка включає також і заперечення формули проблеми, знайдено вивід формули false, а отже, теорема 3 доведена у теорії першого порядку CoreRBACTheory.

5. Використання засобів автоматичного доведення теорем для генерації авторизаційних рішень

Одним з компонентів будь-якої інфраструктури контролю доступу є модуль прийняття авторизаційних рішень (Authorization Decision Point). Головне завдання цього модуля: на базі поточного стану інфраструктури контролю доступу здійснити авторизаційне рішення або, іншими словами, давати відповідь на запитання: чи дозволити деякому користувачу виконати певну операцію над певним ресурсом? Тут запропоновано підхід, за яким авторизаційні рішення генеруються за допомогою засобів автоматичного доведення теорем.

Розглянемо рис. 3, що демонструє принцип роботи модуля.

Розглянемо покроково алгоритм запропонованого підходу:

1. Користувач відсилає системі запит на виконання певної операції над деяким ресурсом. Цей запит перехоплюється модулем Authorization Module.
2. Дані, що містяться в запиті, трансформуються у теорему авторизаційного твердження. На їхній базі формується файл проблеми Problem.tstp.
3. Файл проблеми Problem.tstp доповнюється аксіомами, що відповідають політикам контролю доступу. Файл проблеми подається на вхід ПЗ EProver.
4. На основі вихідного файлу ПЗ EProver формується авторизаційна відповідь.

Розглянемо докладніше структуру файлів Problem.tstp та Result.tstp на прикладах моделей

контролю доступу CoreRBAC та HierarchicalCoreRBAC.

В обох випадках файл проблеми Problem.tstp буде складатися з таких частин:

– аксіоми моделі контролю доступу;
– аксіоми поточного стану моделі контролю доступу;
– теорема авторизаційного рішення.

Відповідно вихідний файл ПЗ EProver Result. tstp міститиме відповідь на запитання, чи може бути теорема авторизаційного рішення доведеною в аксіоматичній теорії першого порядку, що задана у файлі Problem.tstp?

5.1. Використання засобів автоматичного доведення теорем для генерації авторизаційних рішень у випадку моделей контролю доступу на базі ролей Core RBAC та Hierarchical RBAC

Розглянемо стан моделі контролю доступу CoreRBAC:

CoreRBAC = (U, Ro, O, Re, URo, RoORe), де
U = {1, 2},
Ro = {3},
O = {4},
Re = {5},
URo = {(1,3)},
RoORe = {(3,4,5)}.

Розглянемо аксіоми і теореми, що будуть міститися в файлі проблеми Problem.tstp.

Секція аксіом контролю доступу міститиме аксіоми теорії першого порядку CoreRBACTheory.

Секція аксіом поточного стану моделі контролю доступу міститиме аксіоми, що однозначно задаватимуть множини U, Ro, O, Re, URo, RoORe. На жаль, мова TSTP не має механізмів задання множин, тому відповідні множини поточного статусу будуть задані як предикати-індикатори та матимуть вигляд:

$\forall x, U(x) \leftrightarrow (x = 1 \vee x = 2)$;
$\forall x, Ro(x) \leftrightarrow (x = 3)$;
$\forall x, O(x) \leftrightarrow (x = 4)$;
$\forall x, Re(x) \leftrightarrow (x = 5)$;
$\forall x, y, URo(x, y) \leftrightarrow (x = 1 \wedge y = 3)$;
$\forall x, y, z, ROORe(x, y, z) \leftrightarrow (x = 3 \wedge y = 4 \wedge z = 5)$.

Розглянемо тепер, як буде утворюватися файл проблеми Problem.tstp для стану моделі, що описаний. Оскільки в цьому розділі розглядатимемо кілька видів авторизаційних тверджень, спочатку звернемо увагу на аксіоматичну частину файлу проблеми, а потім, залежно від авторизаційного твердження, проаналізуємо різні варіанти виразів проблеми.

Як зазначалося, аксіоматична частина файлу проблеми Problem.tstp складається з двох частин: аксіоми теорії першого порядку CoreRBACTheory та аксіоми поточного стану моделі контролю доступу.

Частина аксіом теорії першого порядку CoreRBACTheory файлу проблеми Problem.tstp матиме такий вигляд:

```
% axiom #1
fof(a1, axiom, ![C]:(~(o(C) | re(C) | ro(C)) | ~(u(C) | re(C) | ro(C)) | ~(o(C) | u(C) | ro(C)) | ~(o(C) | re(C) | u(C)) & (o(C) | re(C) | ro(C) | u(C)))).
% axiom #2
fof(a2, axiom, ![X,Y,Z] : ((u(X) & o(Y) & re(Z) & uore(X,Y,Z)) <=> ? [RO] : (u(X) & o(Y) & re(Z) & ro(RO) & uro(X,RO) & roore(RO,Y,Z)))).
% axiom #3
fof(a3, axiom,![U,RO]:((u(U) & ro(RO) & uro(U,RO))<=>(u(U) & ro(RO) &! [O,RE] :((o(O) & re(RE) & roore(RO,O,RE))=>(o(O) & re(RE) & uore(U,O,RE))))))).
% axiom #4
fof(a4, axiom, ! [RO,O,RE]:((ro(RO) & o(O) & re(RE) & roore(RO,O,RE))<=>(ro(RO) & o(O) & re(RE) &! [U] :((u(U) & uro(U,RO))=>(u(U) & uore(U,O,RE))))))).
```

Частина аксіом поточного стану моделі контролю доступу матиме вигляд:

```
fof(p1, axiom, u(X) <=> (X=1 | X=2)).
fof(p2, axiom, ro(X) <=> X=3).
fof(p3, axiom, o(X) <=> X=4).
fof(p4, axiom, re(X) <=> X=5).
fof(p5, axiom, uro(X,Y) <=> (X=1&Y=3)).
fof(p6, axiom, roore(X,Y,Z) <=> (X=3&Y=4&Z=5)).
```

Далі розглянемо приклади авторизаційних тверджень і те, яким чином можна використати запропонований підхід для визначення, чи є вони правильні чи хибні.

У запропонованому прикладі процес перевірки правильності чи хибності певного авторизаційного твердження складається з двох кроків:

– спроба знайти вивід формули авторизаційного твердження;
– спроба знайти вивід формули заперечення авторизаційного твердження.

Тож, якщо система знайде виведення формули на першому кроці, то авторизаційне твердження є правильним, якщо на другому – хибним.

Розглянемо наступне авторизаційне твердження: користувач 2 може виконати операцію 4 над ресурсом 5. Цьому твердженню буде відповідати формула UORe (2, 4, 5), а його запереченню – формула ¬UORe(2, 4, 5).

Формулі авторизаційного твердження та її запереченню відповідатимуть такі вирази мовою TSTP:

```
1) fof(c1, conjecture, uore(2,4,5));
2) fof(c1, conjecture, ~uore(2,4,5)).
```

Зауважимо, що у другому випадку ПЗ EProver знайде шлях виводу формули, а у першому – ні, що означатиме: авторизаційне твердження є хиб-

ним, тобто користувач 2 не має права виконувати операцію 4 над ресурсом 5.

Крім перевірки авторизаційних тверджень, запропонований підхід дає змогу досліджувати властивості поточного стану моделі контролю доступу. В цьому випадку властивість повинна бути задана як формула теорії першого порядку CoreRBACTheory і додана у файл проблеми Problem.tstp, що вже містить аксіоми теорії та аксіоми поточного стану. Розглянемо на прикладі цю можливість.

Нехай нам потрібно перевірити, чи існують така операція O та ресурс P, що користувач 2 може виконати операцію O над ресурсом P. Формально, ця властивість може бути представлена формулою теорії першого порядку CoreRBACTheory:

$$\exists x, y, O(x) \wedge Re(y) \wedge UORe(2, x, y).$$

Відповідно до запропонованого підходу потрібно перевірити на виводність цю формулу та її заперечення. Отже, ми повинні послідовно перевірити наступні дві проблеми, записані мовою TSTP:

1) fof(c1, conjecture, ? [Y,Z] : (o(Y) & re(Z) & uore(2,Y,Z)));
2) fof(c1, conjecture, ~? [Y,Z] : (o(Y) & re(Z) & uore(2,Y,Z))).

У першому випадку ПЗ EProver відповість, що формула не виводиться, у другому – що виводиться, а це означатиме, що задана властивість не виконується у наведеному поточному стані.

Слід зазначити, що аналогічним чином можна досліджувати і модель HierarchicalRBAC, доповнивши її аксіомами (13т), (14т), (15т), (16т) та (17т). Це означає, що частину аксіом моделі контролю доступу файлу проблеми Problem.tstp потрібно доповнити такими виразами мови TSTP:

% axiom 13t

fof(axiom, 13t, ! [ROA, ROB] : ((ro(ROA) & ro(ROB) & roh(ROA, ROB)) ⇔ ! [O, RE] : (ro(ROA) & ro(ROB) & (o(O) & re(RE) & roore(ROB, O, RE)) => (o(O) & re(RE) & roore(ROA, O, RE)))).

% axiom 14t

fof(axiom, 14t, ! [ROA,ROB] : (ro(ROA) & ro(ROB) & (roh(ROA, ROB) | ? [ROC] : (ro(ROC) & rohi(ROA, ROC) & rohi(ROC, ROB))))).

% axiom 15t

fof(axiom, 15t, ! [RO] : (ro(RO) => rohi(RO, RO))).

% axiom 16t

fof(axiom, 16t, ! [RO, O, RE] : (ro(RO) & o(O) & re(RE) & roorei(RO, O, RE)) ⇔ ? [ROA] : (ro(RO) & o(O) & re(RE) & ro(ROA) & rohi(RO, ROA) & roore(ROA, O, RE))).

$$\forall u,o,re\ (U(u) \wedge O(o) \wedge Re(re) \wedge u\ (o)(...) \xrightarrow{UORE} re) \leftrightarrow \exists ro\ (U(u) \wedge O(o) \wedge Re(re) \wedge Ro(ro) \wedge u \xrightarrow{URO} ro \wedge ro\ (o)(...) \xrightarrow{ROORE} re) \quad (17т)$$

% axiom 17t

fof(axiom, 17t, ! [U, O, RE] : (u(U) & o(O) & re(RE) & uorei(U, O, RE)) ⇔ ? [RO] : (u(U) & o(O) & re(RE) & ro(RO) & uro(U, RO) & roore(RO, O, RE))).

Також у секцію аксіом поточного стану необхідно додати аксіому, що задавала б відношення RoH моделі контролю доступу Hierarchical RBAC. Наприклад, якщо відношення RoH будується як {(1,2)}, то відповідна аксіома повинна мати такий вигляд:

$$\forall x, y, RoH(x, y) \leftrightarrow (x = 1 \wedge y = 2),$$

а мовою TSTP аксіома буде записана так:

fof(axiom, rohaxiom, ! [X, Y] :
: (roh(X, Y) ⇔ (X = 1 & Y = 2))).

Далі генерувати авторизаційні твердження і досліджувати властивості поточного стану моделі контролю доступу можна так само, як і у випадку CoreRBACTheory.

6. Висновки

Розглянуто проблеми формалізації моделей контролю доступу стандарту *ANSI-INCITS 359-2004 Role Based Access Control* як аксіоматичні теорії першого порядку та досліджено їхні властивості за допомогою засобів автоматичного доведення теорем. Також запропоновано використання засобів автоматичного доведення теорем для автоматичної генерації авторизаційних рішень.

Один з можливих шляхів вдосконалення запропонованих підходів є формулювання аксіом відповідних теорій першого порядку у вигляді Хорнівських диз'юнктивних форм, що дасть змогу використовувати DLS-резолюцію, доводячи властивості теорій й авторизаційних тверджень, що значно підвищить швидкодію системи контролю доступу.

1. American National Standards Institute, Inc., Role Based Access Control, ANSI INCITS 359–2004.
2. Ferraiolo D. F. Role Based Access Control / D. F. Ferraiolo, D. R. Kuhn. – 15th National Computer Security Conference, 1992.
3. Ferraiolo D. F. Role Based Access Control : Features and Motivations / D. F. Ferraiolo, J. Cugini, D. R. Kuhn. – Computer Security Applications Conference, 1995.
4. Sandhu R. S. Role-Based Access Control Models / R. S. Sandhu, E. J. Coyne, H. L. Feinstein, C. E. Youma. – IEEE Computer 29(2) : 38–47, IEEE Press, 1996.
5. National Computer Security Center, A Guide to Understanding Discretionary Access Control in Trusted Systems, Library No. S–228, 576, 1987.

6. Плиско В. Е. Математическая логика. Курс лекций / В. Е. Плиско. – Режим доступу: http://lpcs.math.msu.su/~plisko/matlog.
7. Режим доступу: http://eprover.org
8. Режим доступу: http://tptp.org

A. Kolyadenko

USAGE OF AUTOMATIC THEOREM PROVERS FOR RBAC MODELS RESEARCH AND GENERATING OF AUTHORIZATION DECISION STATEMENTS

Usage of automatic theorem provers for RBAC models research was considered. The first-order theory representations of ANSI-INCITS 359-2004 RBAC models were offered. The approach of generating of authorization decision statements using automatic theorem prover EProver was offered.

УДК 681.3.06

Шкільняк О. С.

СЕКВЕНЦІЙНІ ЧИСЛЕННЯ КОМПОЗИЦІЙНО-НОМІНАТИВНИХ МОДАЛЬНИХ І ТЕМПОРАЛЬНИХ ЛОГІК

На основі інтегрованого інтенсіонально-екстенсіонального підходу до побудови логічних та програмних систем досліджено композиційно-номінативні модальні та темпоральні логіки номінативних рівнів. Для зазначених логік збудовано числення секвенційного типу. Для таких числень доведено теореми коректності та повноти.

Останім часом для специфікації програм і моделювання складних динамічних систем ефективно використовуються модальні та темпоральні логіки. Можливості модальних логік та композиційно-номінативних логік квазіарних предикатів поєднують композиційно-номінативні модальні логіки (КНМЛ) [1]. Важливим класом композиційно-номінативних модальних логік є транзиційні КНМЛ, окремим випадком яких є загальні та темпоральні КНМЛ. Центральним поняттям КНМЛ є поняття композиційно-номінативної модальної системи (КНМС) [1]. В роботі [2] на основі інтегрованого інтенсіонально-екстенсіонального підходу [3] запропоноване спеціальне уточнення поняття КНМС для логік реномінативного та кванторного рівнів, досліджені семантичні властивості транзиційних та темпоральних КНМЛ. Для цих логік запропоновано числення секвенційного типу, доведено коректність та повноту таких числень пропозиційного рівня.

В цій роботі продовжується дослідження секвенційних числень транзиційних та темпоральних КНМЛ номінативних рівнів. Розглядається відношення логічного наслідку для множин формул таких логік. На основі властивостей цього відношення будуються секвенційні числення для загальних транзиційних (ТМЛ) та темпоральних КНМЛ реномінативного та кванторного рівнів, для таких числень доводяться теореми коректності та повноти.

Поняття, які тут не визначаються, будемо тлумачити за роботами [1, 2].

1. Відношення логічного наслідку для множин формул КНМЛ

Для КНМЛ номінативних рівнів поняття логічного наслідку для множин специфікованих станами формул визначаємо так:

Δ є логічним наслідком **Γ** в КНМС ***M***, якщо для всіх $d \in {}^{V}A$ із того, що $\Phi_{\alpha}(d \cap {}^{V}A_{\alpha}) = T$ для всіх