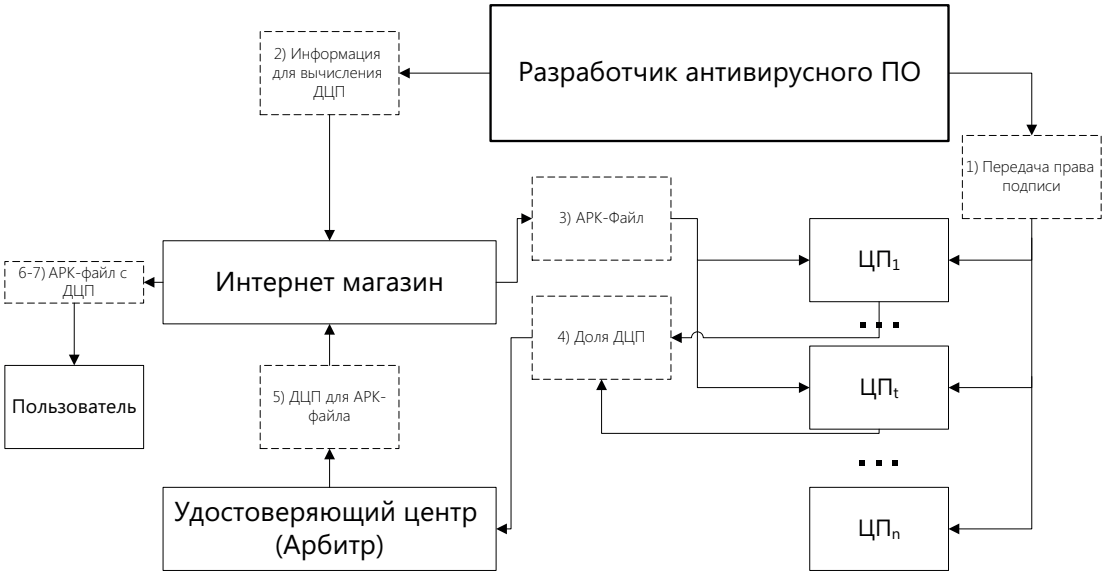

Рис. 1. Схема взаимодействия участников

(1,2 ГГц). В ходе тестирования, согласно данным приложения Android Assistant, использовалось только одно ядро многоядерных процессоров. Тестирование производилось на APK-файлах 5,3 и 76 Мбайт. В таблице для каждого из файлов указано время проверки 10000 ДЦП.

Процессор	**Файл 5,3 Мбайт**	**Файл 76 Мбайт**
Exynos 4412	97,4 с	1241,56 с
Exynos 4210	143,21 с	1785,56 с

ЛИТЕРАТУРА

1. *Толюпа Е. А.* Механизм антивирусной защиты на базе (n, t)-пороговой ДЦП с арбитром // МАИС. 2014 (в печати).
2. *Mambo M., Usuda K., and Okamoto E.* Proxy signatures for delegating signing operation // Proc. of 3rd ACM Conference on Computer and Communications Security (CCS'96). ACM Press, 1996. P. 48–57.

УДК 004.94

ДП-МОДЕЛЬ МАНДАТНОГО УПРАВЛЕНИЯ ДОСТУПОМ С КОНТРОЛЕМ ЦЕЛОСТНОСТИ СУБД MySQL

Д. В. Чернов

Работа посвящена разработке механизмов мандатного контроля целостности в мандатной ДП-модели СУБД MySQL. Вводятся основные элементы модели (решётка уровней целостности сущностей, функции избирательности контроля целостности, уровней целостности сущностей и т. д.), с помощью которых обеспечивается мандатный контроль целостности; описываются некоторые правила преобразования состояний системы; определяются состояния системы, в которых не происходит нарушения целостности, и формулируются условия, которые необходимы, чтобы система оставалась в этих состояниях.

Ключевые слова: *управление доступом, мандатный контроль целостности, информационные потоки, формальные модели безопасности.*

Рассматривается расширение мандатной ДП-модели MySQL [1], включающее в себя мандатный контроль целостности, основанный на модели Биба [2, 3]. Целью мандатного контроля целостности является предотвращение возможности получить доступ на запись сущности с высоким уровнем целостности субъект-сессией с низким уровнем целостности. Разрешается изменять сущность только таким субъект-сессиям, чьи уровни целостности не меньше уровня целостности сущности. Помимо этого, накладывается запрет на вызов процедур, целостность которых ниже целостности пользователя, от имени которого предписано их выполнение. В противном случае решение о возможности выполнить запрос процедуры, который, ввиду его низкой целостности, мог быть изменён третьим лицом, будет выполняться на основе более высокой целостности пользователя. Подобный сценарий может нарушить целостность других сущностей, что, несомненно, является недопустимым.

Для расширения модели мандатным контролем целостности вводятся следующие дополнительные элементы (недостающие специальные термины и обозначения из теории ДП-моделей см. в [3]):

1. Решётка упорядоченных уровней целостности сущностей $(LI, \leqslant)$.

2. Функция $HLSI : O \cup (C \setminus c_0) \to \{\texttt{true}, \texttt{false}\}$ определяет наличие проверок мандатного контроля целостности при доступе к сущности. Если $HLSI(e) = \texttt{true}$, то доступ к сущности e осуществляется с учётом проверок мандатного контроля целостности, в противном случае — без него. При этом значение функции $HLSI(e)$ наследуется всеми сущностями, которые иерархически подчинены e, т. е.

$$\forall e' < e \; (HLSI(e') = HLSI(e)).$$

Предполагается, что если $\exists e \in O \cup C \; (HLSI(e) = \texttt{true})$, то для всякого $e' \in O_p \cup \cup O_t$ выполняется $HLSI(e') = \texttt{true}$. То есть если доступ хотя бы к одному объекту или контейнеру осуществляется с учётом мандатного контроля целостности, то так же осуществляется доступ ко всем процедурам и триггерам.

3. Функция $id_e : O \cup C \to L \cup \{\varnothing\}$ определяет уровень целостности сущности-объекта или контейнера таким образом, что выполняются следующие условия для $e \in O \cup C$:

— если $HLSI(e) = \texttt{false}$, то $id_e(e) = \varnothing$;
— если $HLSI(e) = \texttt{true}$ и $\neg\exists e' > e (id_e(e') \neq \varnothing)$, то $id_e(e') \neq \varnothing$;
— если $id_e(e) \neq \varnothing$ и $\exists e' > e (id_e(e') \neq \varnothing)$, то $id_e(e) > id_e(e')$.

4. Функция $id_s : U \to L \cup \{\varnothing\}$ определяет уровень целостности учётной записи пользователя.

Предполагается, что если значение функции id_e определено хотя бы для одной сущности e, то функция id_s определена для всех пользователей, т. е.

$$(\exists e \in O \cup C \; (id_e(e) \neq \varnothing)) \Rightarrow (\forall u \in U \; (id_s(u) \neq \varnothing)).$$

5. Функция $ih_e : O \cup C \to L \cup \{\varnothing\}$ задаёт иерархические уровни целостности сущностей. Определена для $e \in O \cup C$ следующим образом:

— если $id_e(e) \neq \varnothing$, то $ih_e(e) = id_e(e)$;
— иначе
  — если $HLSI(e) = \texttt{true}$, $\exists e' > e \; (HLSI(e') = \texttt{true}, id_e(e') \neq \varnothing, \neg\exists e'' \; (e' > e'' > e, id_e(e'') \neq \varnothing))$, то $ih_e(e) = id_e(e')$;
  — если $HLSI(e) = \texttt{false}$, то $ih_e(e) = \varnothing$.

Корректность определения функции ih_e обосновывает следующее

Утверждение 1. Если $HLSI(e) = \texttt{true}$ и $id_e(e) = \varnothing$, то

$$\exists e' > e \ (HLSI(e') = \texttt{true}, id_e(e') \neq \varnothing).$$

В качестве примера приведём следующие правила преобразования состояний расширенной модели (таблица).

Правило	Исходное состояние G	Результирующее состояние G'
$access\_read(s, e)$	$s \in S$, $e \in DB \cup TAB \cup COL$; если $HLSI(e) = \texttt{true}$, то $\neg\exists e' \in O \cup C (HLSI(e') = \texttt{true}$, $ih_e(e') > ih_e(e), (s, e', write_a) \in A)$	$A' = A \cup \{(s, e, read_a)\}$, $F' = F \cup \{(e, s, write_m)\}$
$access\_write(s, e)$	$s \in S$, $e \in DB \cup TAB \cup COL$	если $HLSI(e) = \texttt{true}$, то $id_s(user(s)) \geqslant ih_e(e)$ и $\neg\exists e' \in O \cup C (HLSI(e') = \texttt{true}, ih_e(e') < ih_e(e)$, $(s, e', read_a) \in A)$; $A' = A \cup \{(s, e, write_a)\}$, $F' = F \cup \{(s, e, write_m)\}$
$execute\_proc(s, p)$	$s \in S$, $p \in O_p$; если $execute\_as(p) = as\_owner$, то $w = owner(p)$, иначе $w = user(s)$, $id_s(w) \leqslant ih_e(p)$	$A' = A \cup \{(s, p, execute_a)\}$; если $execute\_as(p) = as\_owner$, то положим $user(s) = owner(p)$, выполним последовательно $G = G_0 \vdash_{op_1} G_1 \vdash \ldots \vdash_{op_k} G_k = G'$, где $(op_1, \ldots, op_k) \in operations(p)$. Вернём начальное значение $user(s)$

Будем говорить, что в состоянии системы не происходит нарушения целостности, если в нём выполняются следующие условия:

— $\neg\exists(e_1, e_2, write_m) \in F$, где $e_1, e_2 \in E$ и $i_e(e_1) < i_e(e_2)$;
— $\neg\exists(s, p, execute_a) \in A$, где $s \in S$, $p \in O_p$ и $i_e(p) < i_s(s)$.

Говорим также, что в системе не происходит нарушения целостности, если не происходит нарушения целостности во всех её состояниях на всех траекториях функционирования системы.

Теорема 1. Пусть в начальном состоянии системы не происходит нарушения целостности и начальные множества доступов и информационных потоков пустые. Тогда в системе не происходит нарушения целостности.

ЛИТЕРАТУРА

1. *Колегов Д. Н., Ткаченко Н. О., Чернов Д. В.* Разработка и реализация мандатных механизмов управления доступом в СУБД MySQL // Прикладная дискретная математика. Приложение. 2013. № 6. С. 62–67.
2. *Biba K. J.* Integrity Considerations for Secure Computer Systems. Technical Report MTR-3153. MITRE Corp., 1975.
3. *Девянин П. Н.* Модели безопасности компьютерных систем. Управление доступом и информационными потоками: учеб. пособие для вузов. М.: Горячая линия-Телеком, 2012. 320 с.