УДК 004.94

О РАЗРАБОТКЕ МЕХАНИЗМА МАНДАТНОГО УПРАВЛЕНИЯ ДОСТУПОМ В СУБД MySQL НА ОСНОВЕ SELinux

Н. О. Ткаченко

Рассматривается подход к разработке механизма управления доступом в системе управления базами данных (СУБД) MySQL на основе SELinux.

В СУБД MySQL [1] используется дискреционный механизм управления доступом, реализованный следующим образом. Имеется база данных (БД) *mysql*, которая содержит таблицы, где хранится различная служебная информация. Записи, отражающие права доступа субъектов к сущностям, находятся в следующих таблицах:

— *user*;
— *db*;
— *host*;
— *tables_priv*;
— *columns_priv*;
— *procs_priv*.

Таблица *user* хранит глобальные права доступа субъектов к сущностям. Глобальные права распространяются на все сущности СУБД. Таблицы *db* и *host* определяют права для БД. В остальных таблицах находятся права, определяющие доступ субъекта к таким сущностям, как таблицы и столбцы.

Процесс управления доступом начинается с проверки наличия глобальных прав. Если они присутствуют, то доступ разрешается. В случае отсутствия каких-либо прав у субъекта в таблице *user* последовательно просматриваются оставшиеся таблицы.

В СУБД MySQL отсутствуют механизмы мандатного управления доступом. В то же время в известных СУБД (например, Oracle) имеется возможность реализации политики мандатного управления доступом. В СУБД MySQL это может быть реализовано с использованием механизма безопасности SELinux [2].

В системе SELinux политика безопасности управления доступом компьютерной системы (КС) задаётся набором правил, описывающих права доступа субъектов к сущностям на основе их контекста безопасности, в виде единого конфигурационного файла или набора модулей [3, 4]. Под контекстом безопасности понимается набор атрибутов, связанных с сущностью КС и имеющих вид $user : role : type[: level]$, где

— *user* — атрибут-пользователь в системе SELinux, ассоциированный с одним или более пользователем КС;
— *role* — атрибут-роль в системе SELinux, ассоциированная с одним или более типом, к которым пользователь SELinux имеет доступ;
— *type* — атрибут-тип в системе SELinux, определяющий возможные виды доступа сущностей данного типа при использовании мандатного механизма `Type Enforcement`;
— *level* — атрибут-уровень безопасности в системе SELinux при использовании мандатных механизмов `Multy-Level Security` или `Multy-Category Security`.

Система SELinux состоит из следующих частей:

1) *Object Manager* (OM) — служит посредником при принятии решения о разрешении/запрете доступа субъекта к объекту;
2) *Access Vector Cache* (AVC) — необходим для оптимизации работы системы SELinux;

3) сервер SELinux — применяется для создания ответа на основе запроса и политики SELinux.

При попытке субъекта получить доступ к сущности ОМ составляет запрос и отправляет его к AVC. AVC принимает запрос от ОМ, и если ранее поступал такой запрос, то AVC находит его в своей базе и отправляет ответ ОМ, иначе запрос перенаправляется серверу SELinux. Сервер SELinux при поступлении запроса находит соответствующее правило в политике безопасности и отправляет ответ AVC, который, в свою очередь, запоминает его и перенаправляет ОМ.

ОМ реализован на уровне ядра ОС GNU/Linux, при этом SELinux предоставляет средства (библиотека `libselinux`), реализующие ОМ на уровне пользовательского приложения.

С использованием данной возможности для СУБД MySQL (версия 5.5.16) разработан прототип модуля, задающий мандатную политику управления доступом на основе механизма `Type Enforcement`, и элемент ОМ, реализующий её на пользовательском уровне, а также определены контексты безопасности для основных сущностей СУБД MySQL.

В СУБД MySQL для присвоения сущностям контекстов безопасности использованы следующие таблицы БД *mysql*:

— *SEDB*;
— *SETable*;
— *SEColumn*.

В этих таблицах сопоставляются контексты безопасности и объекты *DB*, *Table*, *Column* СУБД MySQL. В служебной таблице *user* БД *mysql* создан столбец `sec_context`, задающий контексты безопасности субъектам. Для задания контекста безопасности записям необходимо в пользовательской таблице создать столбец `sec_context`.

Одним из типичных способов реализации элемента ОМ является добавление хук-функций в исходный код, содержащий функции управления доступом субъектов к сущностям КС. В СУБД MySQL такими функциями являются:

— *bool check_access*() — функция, реализующая управление доступом на основе табличных данных *user*, *db*, *host*;
— *bool check_grant*() — функция, реализующая управление доступом на основе табличных данных *table_priv*;
— *bool check_grant_column*() — функция, реализующая управление доступом на основе табличных данных *column_priv*.

В момент реализации субъектом доступа к сущности выполняется проверка глобальных прав доступа, при этом вызывается функция *check_access*(), передающая управление хук-функции. Последняя взаимодействует с сервером SELinux и, в зависимости от ответа, выполняет действия по запрету или разрешению доступа.

Таким образом, основными этапами реализации политики мандатного управления доступом в СУБД MySQL на основе механизма SELinux являются:

1) задание контекста безопасности для каждой сущности СУБД MySQL;
2) разработка модуля политики безопасности;
3) создание функций, реализующих элемент *ОМ* для СУБД MySQL;
4) замена функций, реализующих управление доступом в СУБД MySQL на функции, вызывающие механизмы SELinux.

ЛИТЕРАТУРА

1. *Hinz S., DuBois P., Stephens J., et al.* MySQL 5.5 Reference Manual [Электронный ресурс]. Режим доступа: http://dev.mysql.com/doc/refman/5.5/en/index.html
2. *Haines R.* The SELinux Notebook — The Foundations. 2nd Edition [Электронный ресурс]. Режим доступа: http://www.freetechbooks.com/efiles/selinuxnotebook/The_SELinux_Notebook_Volume_1_The_Foundations.pdf
3. *Smalley S.* Configuring the SELinux Policy [Электронный ресурс]. Режим доступа: http://www.nsa.gov/research/_files/selinux/papers/policy2.pdf
4. *Loscocco P. A., Smalley S. D.* Meeting Critical Security Objectives with Security-Enhanced Linux [Электронный ресурс]. Режим доступа: http://www.nsa.gov/research/_files/selinux/papers/ottawa01.pdf

УДК 004.94

О МОДЕЛЯХ ЛОГИЧЕСКОГО УПРАВЛЕНИЯ ДОСТУПОМ НА ОСНОВЕ АТРИБУТОВ

Д. В. Чернов

Доклад посвящен обзору основных работ по моделям логического управления доступом на основе атрибутов, или, иначе, атрибутного управления доступом (Attribute Based Access Control) в компьютерных системах (КС). При таком виде управления доступом предоставление субъекту права доступа к сущности происходит только в том случае, если значения атрибутов субъектов и сущностей позволяют субъекту предоставить данный доступ к сущности. Как правило, атрибутное управление доступом рассматривается как отдельный вид управления доступом наряду с дискреционным, мандатным и ролевым. Вместе с тем КС с управлением доступом на основе атрибутов могут использовать в качестве последних типы, уровни безопасности и роли, включая в себя соответственно отдельные дискреционные, мандатные и ролевые механизмы. В общем случае механизмы функционирования атрибутного управления доступом характерны для систем с мандатным управлением доступом [1].

Атрибутное управление доступом является новым и перспективным видом политик логического управления доступом и информационными потоками в КС. Большинство работ по моделям атрибутного управления доступом (например, [2]) ориентированы на реализацию или оптимизацию подсистемы управления доступом; в них используются оригинальные определения элементов и механизмов защиты, а используемый математический аппарат часто недостаточен для анализа условий нарушения безопасности КС и формального обоснования методов и требований их защиты. В то же время известны модели атрибутного управления доступом, например [3], исследующие вопросы теоретического анализа безопасности.

Исторически первой моделью атрибутного управления доступом может считаться модель типизированной матрицы доступа (ТМД), в которой с каждым объектом системы ассоциирован атрибут-тип. В настоящее время предложено несколько подходов к управлению доступом на основе атрибутов.

В [4] подробно рассматривается модель с динамической ролью. Предполагается, что в системе имеется некоторое количество ролей с заранее определенными правами. К системе имеют доступ неограниченное количество пользователей, которым необходимо приписывать какие-то роли. При этом роли могут меняться с течением времени. Для определения роли пользователя в текущий момент времени используются значе-