

Методика оценки полноты покрытия обращений субъектов к объектам

Оглавление

Введение.....	1
Основные принципы.....	1
Типы субъектов и объектов.....	2
Операции и связанные с ними особые ситуации.....	3

Введение

Данный документ описывает методику оценки полноты для тестов обращений субъектов к объектам операционной системы (ОС). Описываемая методика основана на формальной модели политик безопасности ОС и отображении понятий этой модели в реализацию моделируемой ОС.

Основные принципы

Представленная в данном документе методика оценки полноты тестов определяет критерий полноты тестирования. Этот критерий выделяет два вида ситуаций.

- Ситуации, связанные с использованием в качестве аргументов тестируемых операций объектов и субъектов ОС, относящихся к специфическим подтипам.
Подтипы объектов и субъектов выделяются на основе их классификации, присутствующей в формальной модели политик безопасности ОС, на основе возможных значений атрибутов и возможных связей с другими сущностями системы, которые могут повлиять на выполнение тестируемых операций с точки зрения формальной модели политик безопасности, а также на основе разнообразных способов реализации этих атрибутов и связей в рамках тестируемой ОС.
- Ситуации, связанные с условиями выполнения самой тестируемой операции, выделяемыми в рамках формальной модели политик безопасности ОС.
В качестве таких условий рассматриваются все охранные условия, перечисленные в предусловии тестируемой операции в формальной модели политик безопасности. В тестах должны проверяться все ситуации, связанные с нарушением одного из охранных условий и выполнением всех остальных. Если есть несколько способов нарушить охрannое условие (если, например, оно сформулировано в виде конъюнкции нескольких простых условий), должны быть проверены ситуации, полученные всеми этими способами. Для некоторых комбинаций нарушений разных охранных условий также должны быть сформированы соответствующие тесты.

При формировании условий и комбинаций условий, для проверки которых должны быть построены тесты, наибольший приоритет отдается условиям, связанным с механизмом мандатного управления доступом. Условия, связанные с ролевым (в виде учета повышенных полномочий пользователя root) и дискреционным (в виде учета стандартных для ОС семейства Linux классов доступа и списков контроля доступа) механизмами управлением доступом также должны быть проверены.

Типы субъектов и объектов

В операциях доступа субъектов к объектам фигурируют сущности следующих типов.

- Субъекты — соответствуют процессам ОС (task).
- Объекты — представляют собой объекты доступа в ОС. В рамках формальной модели политик безопасности объекты могут являться файлами, сокетами или объектами межпроцессного взаимодействия.
- Пользователи — соответствуют пользователям ОС. В рамках формальной модели политик безопасности любой субъект работает от имени некоторого пользователя (эффективного пользователя данного процесса), любой файл имеет владельцем некоторого пользователя.
- Группы — группы пользователей ОС. В рамках формальной модели политик безопасности у каждого пользователя есть первичная группа, к которой он принадлежит, у каждого процесса есть эффективная группа, у каждого файла есть группа-владелец.
- Метки защиты, которыми могут быть помечены субъекты и объекты. В рамках формальной модели политик безопасности каждая метка является непустой строкой.

Для того, чтобы обеспечить необходимую полноту тестов, необходимо, чтобы в них были задействованы следующие подтипы сущностей.

- Субъекты могут быть обычными и привилегированными.
Кроме того, процессы могут работать от имени обычного пользователя или root'a (использование этой классификации позволяет проверить условия, касающиеся управления доступом на основе ролей).
Привилегированный субъект тестируемой ОС — это процесс, имеющий полномочия CAP_MAC_OVERRIDE или CAP_MAC_ADMIN. Предполагается, что в системе все процессы, запускаемые от имени root'a, имеют такие полномочия, и наоборот, только root'овые процессы могут их иметь, поэтому смешанных категорий процессов не возникает (если получится добиться для процесса одного из указанных полномочий, не используя полномочия root'a, это само по себе будет брешью в защите системы.).
При этом имеет смысл рассматривать отдельно процессы с одним из указанных полномочий или с обоими сразу.
Кроме того, полезно посмотреть на процессы, получившие пользователя не от родителя, а косвенно — через su или через setuid.
В результате важные подтипы процессов следующие.
 - Процесс, запущенный от имени root;
 - Процесс, запущенный root'ом, но получивший другого эффективного пользователя;
 - Процесс, запущенный от имени обычного пользователя;
 - Процесс, запущенный обычным пользователем, но получивший root в качестве эффективного пользователя.

- Объекты могут быть файлами, сокетами и объектами межпроцессного взаимодействия (объектами IPC).

Файлы могут быть директориями и обычными файлами.

Нужно учитывать, что директория и содержащиеся в ней файлы должны иметь эквивалентные метки защиты (кроме /home). Для реализации всех ситуаций достаточно у каждого тестового пользователя в его home иметь структуру директорий глубины 2 (чтобы проверить возможность доступа к файлам в доступной директории, лежащей в недоступной директории).

Объекты межпроцессного взаимодействия стоит разделить на очереди, семафоры и регионы памяти.

- Пользователи бывают обычные и root. Еще одно различие пользователей — прикрепленные к ним метки защиты. Нужно иметь пользователей со всеми возможными метками защиты. Для некоторых меток защиты нужно иметь 2-х пользователей (чтобы проверить доступ к файлам между разными пользователями с одинаковыми метками), также важно иметь пользователя с максимальной меткой, отличного от root.
- Требования к группам возникают только из специфических ситуаций.
- Метки защиты могут быть обычными и специальными. Обычная метка — это строка длины не более 255 из алфавитных символов. Специальная метка — это строка из одного символа: «*», «^» и «_».
Для проверки разнообразных комбинаций условий, касающихся меток, достаточно иметь две обычные метки (чтобы проверять ограничения на равенство).
При этом, по минимуму, важны следующие ситуации соотношения двух меток.
 - Эквивалентные метки
 - Обычные метки;
 - Специальные метки (всех видов);
 - Неэквивалентные метки
 - Две обычные метки (разные);
 - Две специальные метки (разные);
 - Обычная и специальная метка (все варианты);

Операции и связанные с ними особые ситуации

В модели выделены следующие операции по доступу субъектов к объектам.

- Получение доступа к существующему объекту
- Создание файла или жесткой ссылки
- Удаление файла или жесткой ссылки (переименование файла рассматривается здесь же)
- Чтение содержимого директории (получение имен и др. атрибутов файлов без доступа к самим файлам)

- Создание сокета
- Удаление сокета
- Создание объекта IPC
- Удаление объекта IPC

Для каждой из операций в тестах должны быть проверены следующие ситуации.

- Получение доступа к существующему объекту
 - Для каждого из типов объектов — файлов, сокетов, объектов IPC — проверить все возможные способы передачи объекта процессу.
Файл можно передать через абсолютный или относительный к контексту процесса путь, можно передать дескриптор через сокет, можно передать дескриптор из родительского процесса при запуске.
 - Нужно проверить все возможные комбинации запрашиваемых видов доступа (read, write, exec) с комбинациями типов объектов, меток объекта и процесса и типа пользователя (root, обычный, обычный работающий как root и root, работающий как обычный).
 - Также нужно проверить все возможные комбинации меток объекта и процесса .
 - Нужно проверить все комбинации следующих условий, связанных с механизмом дискреционного управления доступом, для файлов и директорий
Многие из комбинаций являются недостижимыми — маска ACL у файла всегда есть, если есть хотя бы одна запись в ACL пользователей или групп, класс доступа для группы хитро связан с маской и записями в ACL при их наличии.
 - Доступ по владельцу
 - Эффективный пользователь процесса не является владельцем файла
 - Эффективный пользователь процесса является владельцем файла, запрашиваемого вида доступа нет в классе доступа для владельца
 - Эффективный пользователь процесса является владельцем файла, запрашиваемый вид доступа есть в классе доступа для владельца
 - Доступ по маске ACL
 - У файла нет маски ACL
 - Маска есть, запрашиваемой операции в ней нет
 - Маска есть, запрашиваемая операция в ней есть
 - Доступ по пользователю из ACL
 - У файла нет записей в ACL пользователей
 - У файла есть записи в ACL пользователей, но эффективный пользователь процесса там отсутствует

- Эффективный пользователь процесса есть в ACL пользователей файла, но запрашиваемый вид доступа для него отсутствует
- Эффективный пользователь процесса есть в ACL пользователей файла, запрашиваемый вид доступа для него есть, при этом этот пользователь не должен быть владельцем файла или входить в группу-владельца
- Доступ по группе из ACL
 - У файла нет записей в ACL групп
 - У файла есть записи в ACL групп, но группа процесса и все группы пользователя процесса там отсутствуют
 - Группа процесса есть в ACL групп файла, но запрашиваемый вид доступа для нее отсутствует, все группы пользователя процесса там отсутствуют
 - Группа процесса есть в ACL групп файла, запрашиваемый вид доступа для нее есть, все группы пользователя процесса там отсутствуют
 - Группы процесса нет в ACL групп файла, некоторые группы пользователя есть в ACL групп файла, запрашиваемый вид доступа для них отсутствует
 - Группа процесса есть в ACL групп файла, но запрашиваемый вид доступа для нее отсутствует, некоторые группы пользователя есть в ACL групп файла, запрашиваемый вид доступа для них отсутствует
 - Группа процесса есть в ACL групп файла, запрашиваемый вид доступа для нее есть, она должна отличаться от группы-владельца файла, некоторые группы пользователя есть в ACL групп файла, запрашиваемый вид доступа для них отсутствует
 - Группа процесса есть в ACL групп файла, но запрашиваемый вид доступа для нее отсутствует, некоторые группы пользователя есть в ACL групп файла, запрашиваемый вид доступа для некоторых из них есть, при этом такие группы должны отличаться от группы-владельца файла
 - Группа процесса есть в ACL групп файла, запрашиваемый вид доступа для нее есть, она должна отличаться от группы-владельца файла, некоторые группы пользователя есть в ACL групп файла, запрашиваемый вид доступа для некоторых из них есть, при этом такие группы должны отличаться от группы-владельца файла
- Доступ по группе-владельцу
 - Группа-владелец файла не совпадает с группой процесса и не содержит его пользователя
 - Группа-владелец файла совпадает с группой процесса и не содержит его пользователя, запрашиваемого вида доступа нет в классе доступа для группы

- Группа-владелец файла не совпадает с группой процесса и содержит его пользователя, запрашиваемого вида доступа нет в классе доступа для группы
- Группа-владелец файла совпадает с группой процесса и содержит его пользователя, запрашиваемого вида доступа нет в классе доступа для группы
- Группа-владелец файла совпадает с группой процесса и не содержит его пользователя, запрашиваемый вид доступа есть в классе доступа для группы
- Группа-владелец файла не совпадает с группой процесса и содержит его пользователя, запрашиваемый вид доступа есть в классе доступа для группы
- Группа-владелец файла совпадает с группой процесса и содержит его пользователя, запрашиваемый вид доступа есть в классе доступа для группы
- Доступ через всех
 - Запрашиваемый вид доступа отсутствует в классе доступа для других
 - Запрашиваемый вид доступа есть в классе доступа для других
- Наличие и отсутствие доступа на исполнение к содержащей директории, наличие и отсутствие доступа на исполнение к директории, содержащей содержащую
- Отдельно надо проверить работу подсистемы обеспечения целостности. Для этого нужно применить операции доступа на чтение, модификацию и запуск к набору файлов, в котором имеются файлы следующих видов
 - не контролируемые подсистемой обеспечения целостности;
 - корректные контролируемые подсистемой обеспечения целостности (с сохраненным корректным хэш-кодом), но изменяемые (без атрибута `immutable`);
 - некорректные контролируемые подсистемой обеспечения целостности (с сохраненным некорректным хэш-кодом), но изменяемые (без атрибута `immutable`);
 - корректные контролируемые подсистемой обеспечения целостности (с сохраненным корректным хэш-кодом), неизменяемые (с установленным атрибутом `immutable`);
 - некорректные контролируемые подсистемой обеспечения целостности (с сохраненным некорректным хэш-кодом), неизменяемые (с установленным атрибутом `immutable`).

- Создание файла или жесткой ссылки
 - Надо проверить комбинации следующих условий
 - Тип создаваемого объекта — обычный файл, ссылка, директория
 - Метка и тип создающего процесса (root, generic, root-as-generic, generic-as root)
 - Наличие и отсутствие доступа на исполнение и запись к содержащей директории (всех комбинаций), наличие и отсутствие доступа на исполнение к директории, содержащей содержащую
 - Наличие и отсутствие флага setgid на содержащей директории
- Удаление файла или жесткой ссылки

Переименование считается частным случаем этой операции

 - Надо проверить комбинации следующих условий
 - Тип удаляемого объекта — обычный файл, ссылка, директория
 - Все комбинации меток процесса и файла, а также типа процесса (root, generic, root-as-generic, generic-as root)
 - Наличие и отсутствие доступа на исполнение и запись к содержащей директории (всех комбинаций), наличие и отсутствие доступа на исполнение к директории, содержащей содержащую
 - Наличие и отсутствие флага sticky bit на содержащей директории
 - Для переименования — наличие или отсутствие нового имени в содержащей директории
- Чтение содержимого директории
 - Для обычной директории (содержимое имеет те же метки, что и сама директория) надо проверить все комбинации меток читающего процесса и самой директории.
 - Для многоуровневой директории (содержимое может иметь различные метки) надо проверить работу процессов со всеми возможными метками с содержимым, где также представлены файлы со всеми возможными метками.
- Создание сокета
 - Создание сокета процессами со всеми возможными метками, при создании нужно проверить все возможные маски (комбинации read, write и м.б. еще чего-то) для создаваемых сокетов
- Удаление сокета
 - Удаление сокета создавшим его процессом или другим, со всеми возможными метками этого у другого процесса
 - Проверить разные возможные способы передачи сокета процессу
- Создание объекта IPC

- Создание всех типов объектов IPC процессами со всеми возможными метками, при создании нужно проверить все возможные маски (комбинации read, write и м.б. еще чего-то) для создаваемых объектов
- Удаление объекта IPC
 - Удаление объекта создавшим его процессом или другим, со всеми возможными метками этого у другого процесса
 - Проверить разные возможные способы передачи объекта процессу