Слайд 1

Для хранения и обработки информации в современном обществе используются информационные системы. Информация, которой они оперируют, как правило, обладает определённым уровнем конфиденциальности. Это объясняет необходимость систем контроля доступа, позволяющих конкретному субъекту (пользователю) получить допуск к объекту (информации) соответствующего уровня. Получение доступа в некоторых информационных системах осуществляется путём предъявления сертификата открытого ключа — цифрового удостоверения личности субъекта. Использование одного сертификата разрешает субъекту доступ ко всей информации. Для обеспечения доступа субъекта к объектам разного уровня конфиденциальности, а не ко всей информации в целом, необходимо несколько сертификатов соответствующих уровней, доступных субъекту. Ввиду того, что уровней может быть много, то необходима автоматизация процесса выбора сертификата. Поэтому тема данного дипломного проекта является **актуальной**.

В данной дипломной работе предлагается реализация механизма выбора сертификата открытого ключа пользователя на основании его контекста безопасности.

При реализации будет использована инфраструктура открытых ключей (англ. *PKI*), один из принципов построения которой предполагает наличие удостоверяющего центра, выпускающего сертификаты открытых ключей пользователей, тем самым удостоверяя их личность. В каждом из сертификатов в дополнительном атрибуте будет содержаться значение контекста безопасности.

В качестве поставщика метки безопасности будет использоваться SELinux — реализация системы мандатного контроля доступа, которая используется в некоторых дистрибутивах Linux (например, Fedora) вместе с дискреционным механизмом контроля доступа. С помощью специально описанных политик регулируется доступ субъекта (пользователя) к объекту (файлу, директории и т.д.). SELinux может работать в многоуровневом режиме (англ. *MLS*). Этот режим основан на принципе, что субъект может иметь доступ к объекту, если уровень безопасности субъекта соответствует уровню безопасности объекта.

Автоматизация процесса выбора сертификата будет осуществлена с использованием многоэкземплярности директорий ОС семейства Linux — механизма создания независимых копий.

Научная новизна данной работы определяется в выборе сертификата открытого ключа пользователя на основании его контекста безопасности.

В работе показано применение предложенного механизма для аутентификации клиентов СУБД PostgreSQL. Это определяет **практическую значимость** дипломной работы.

Слайд 2

Таким образом, **целью** данной работы является разработка механизма автоматического выбора сертификата пользователя на основании его контекста безопасности. Для достижения поставленной цели были сформулированы следующие **задачи**:

1. Изучить принципы построения инфраструктуры открытых ключей PKI;
2. Исследовать современные средства выбора сертификата открытого ключа;
3. Разработать способ создания сертификатов с контекстом безопасности пользователя;
4. Автоматизировать выбор сертификатов, используя механизм многоэкземплярности;
5. Показать применение разработанного механизма для аутентификации клиентов СУБД PostgreSQL.

Реализация данного механизма произведена на дистрибутиве Linux Fedora 20.

Слайд 3

Инфраструктура открытых ключей (*PKI*, *Public Key Infrastructure*) — набор средств (технических, материальных, людских и т. д.), распределенных служб и компонентов, в совокупности используемых для поддержки криптозадач на основе закрытого и открытого ключей.

В основе PKI лежит использование криптографической системы с открытым ключом и несколько основных принципов:

- закрытый ключ известен только его владельцу;
- удостоверяющий центр создает сертификат открытого ключа, таким образом удостоверяя этот ключ;
- никто не доверяет друг другу, но все доверяют удостоверяющему центру;
- удостоверяющий центр подтверждает или опровергает принадлежность открытого ключа заданному лицу, которое владеет соответствующим закрытым ключом.

PKI реализуется в модели клиент-сервер, то есть проверка какой-либо информации, предоставляемой инфраструктурой может происходить только по инициативе пользователя.

Основные компоненты PKI:

- Удостоверяющий центр (УЦ) является основной структурой, формирующей цифровые сертификаты подчиненных центров сертификации и конечных пользователей. УЦ является главным управляющим компонентом PKI. Он является доверенной стороной.
- Сертификат открытого ключа (чаще всего просто сертификат) — это данные пользователя и его открытый ключ, скрепленные подписью удостоверяющего центра. Выпуская сертификат открытого ключа, удостоверяющий центр тем самым подтверждает, что лицо, поименованное в сертификате, владеет секретным ключом, который соответствует этому открытому ключу.
- Репозиторий — хранилище, содержащее сертификаты и списки отозванных сертификатов (СОС) и служащее для распространения этих объектов среди пользователей.
- Архив сертификатов — хранилище всех изданных когда-либо сертификатов (включая сертификаты с закончившимся сроком действия). Архив используется для проверки подлинности электронной подписи, которой заверялись документы.
- Конечные пользователи — пользователи, приложения или системы, являющиеся владельцами сертификата и использующие инфраструктуру управления открытыми ключами.

Слайд 4

В дипломной работе производится анализ современных подходов к выбору сертификата. Как правило, сертификаты выбираются ПО автоматически на основе дополнений сертификата X509v3:

- назначения ключа
- ограничений сертификата
- политики применения сертификата — нормативных документов, определяющие правила использования сертификата

Аналогов выбора сертификата на основании контекста безопасности найдено не было

Слайд 5

Обычно для создания сертификатов используется библиотека OpenSSL и одноименная утилита командной строки. Стандарт X509v3 предполагает включение в состав сертификата пользовательских дополнений.

Пользовательские дополнения могут быть включены в состав сертификата тремя возможными способами:

- Модификация конфигурационного файла `openssl.conf`;
- Программно:
  - Alias на существующее дополнение (используется структура и методы обработки существующего дополнения, однако новое имеет новое имя и идентификатор);
  - Полная реализация дополнения (реализация структуры, методов обработки).

В работе реализован способ хранения контекста безопасности пользователя путём реализации структуры дополнения `v3_secon`.

Слайд 6

Для утилиты создания сертификата пользователя были определены следующие требования:

1. Возможность создавать закрытый ключ клиента произвольной длины;
2. Создавать запросы на подпись сертификата (CSR) с дополнением `selinuxContext`;
3. Проверка корректности метки безопасности;
4. Выпуск сертификата удостоверяющим центром.

Так как требовалось разработать утилиту максимально быстро, был выбран язык программирования Python. При этом были использована библиотека M2Crypto. Вопросы оптимизации в работе не учитывались.

Слайд 7

Автоматизация выбора сертификата открытого ключа реализована с помощью механизма многоэкземплярности директорий ОС семейства Linux.

Этот механизм предполагает создание копий директорий по определённому признаку. В работе использован механизм многоэкземплярности пользовательских директорий, создающие копии пользовательской директории по текущему контексту безопасности пользователя.

Реализация механизма (создание директории, монтирование экземпляра) — `pam_namespace.so`.

Команды по созданию директорий определяются в скрипте `namespace.init`.

Модуль `pam_namespace` доработан возможностью передавать в скрипт инициализации `namespace.init` значения контекста безопасности пользователя.

При первом входе в систему пользователю создаётся пара закрытый ключ – сертификат, который используется для цифровой подписи. Кроме того, создаётся сертификат, хранящий метку безопасности пользователя на текущем уровне.

Пользователь, имеющий допустимый контекст безопасности, имеет сертификат, в котором содержится значение текущего контекста безопасности.

Слайд 8

Разработанный механизм может быть использован для аутентификации клиентов СУБД PostgreSQL. На слайде приведена схема стенда. На клиентской машине созданы пользователи с определёнными уровнями доступа. На сервере СУБД PostgreSQL им соотносятся пользователи СУБД. Между клиентом и сервером настроено SSL-соединение.

Модуль СУБД `sepgsql` используется для реализации мандатного контроля доступа на основе метки SELinux, был доработан возможностью установки метки безопасности из переданного сертификата клиента.

Модуль `sslinfo`, который содержит набор хранимых процедур для просмотра информации о сертификатах клиента, был дополнен функциями просмотра информации о дополнениях сертификата.

С помощью доработанных модулей реализуется возможность выполнения серверным процессом запросов в том контексте безопасности, который соответствует метки безопасности из клиентского сертификата.

Слайд 9, 10

На следующих слайдах можно убедиться в этом. Пользователь `user2` имеет контекст безопасности `user_u:user_r:user_t:s0-s2`, ему создан сертификат с соответствующей меткой. Пользователь подключается к СУБД PostgreSQL, с помощью функции `ssl_get_extension_by_name()` можно убедиться в наличии дополнения и соответствии значения контекста безопасности клиента; а с помощью функции `sepgsql_getcon()` убедиться в том, что сервер выполняет запросы в том контексте безопасности, который соответствует метке из сертификата.

Слайд 11

Выводы:

- Для хранения контекста безопасности в сертификате X509 было реализовано дополнение X509v3 selinuxContext;
- Разработана утилита pgcert генерации сертификатов с дополнением selinuxContext;
- Для автоматизации процесса выбора сертификата использовался механизм многоэкземплярности директорий;
- Разработанный механизм был адаптирован для аутентификации клиентов СУБД PostgreSQL;
- Расширен функционал библиотек: pam_namespace, M2Crypto;
- Расширен функционал модулей СУБД PostgreSQL: sslinfo, sepgsql;
- Патч для модуля sslinfo был отправлен мировому сообществу PostgreSQL на предмет включения в состав дистрибутива.

Полученные результаты говорят о достижении цели работы.