Реализация механизма автоматического выбора сертификата открытого ключа пользователя на основании его контекста безопасности

Воронин Д.Л., Муравьёв С.К.

17 июня 2014

Цель и задачи дипломного проекта

Цель работы — реализация механизма автоматического выбора сертификата открытого ключа пользователя на основании его контекста безопасности.

Для достижения поставленной цели были поставлены следующие **задачи**:

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

Принципы построения PKI

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

- Закрытый ключ известен только владельцу;
- Удостоверяющий центр выпускает сертификат открытого ключа, т.о. удостоверяя этот ключ;
- Никто не доверяет друг другу, но все доверяют удостоверяющему центру;
- Удостоверяющий центр подтверждает или опровергает принадлежность открытого ключа лицу, которое владеет соответствующим закрытым ключом.

Современные подходы к выбору сертификата

Выбор сертификатов осуществляется автоматически на основе дополнений X509v3:

- назначения ключа — keyUsage;
- ограничений — дополнения Basic Constraints, Name Constraints;
- ППС (политики применения сертификата) — дополнения Certificate Policies, Policy Mappings, Policy Constraints.

OpenSSL

OpenSSL — Криптографический пакет для работы с сертификатами.

Создание дополнений в сертификатах:

- Модификация конфигурационного файла `openssl.conf`;
- Программно:
  - Alias на существующее дополнение;
  - Реализация структуры дополнения.

Реализован способ хранения метки безопасности в дополнении `v3_secon` (программно).

Разработка утилиты создания сертификатов

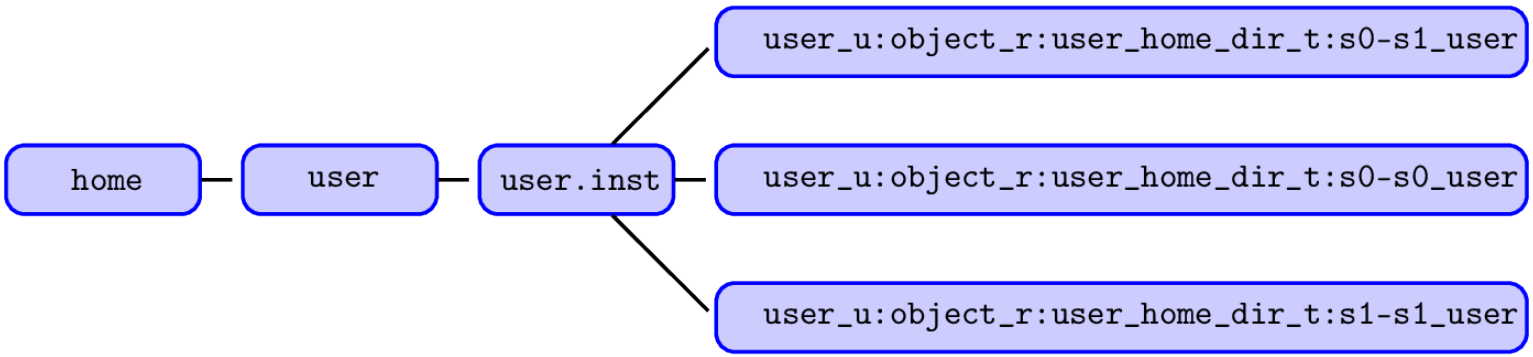
Требования:

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.

Реализована утилита на языке Python `pgcert` с использованием библиотеки M2Crypto.

Автоматизация выбора сертификата

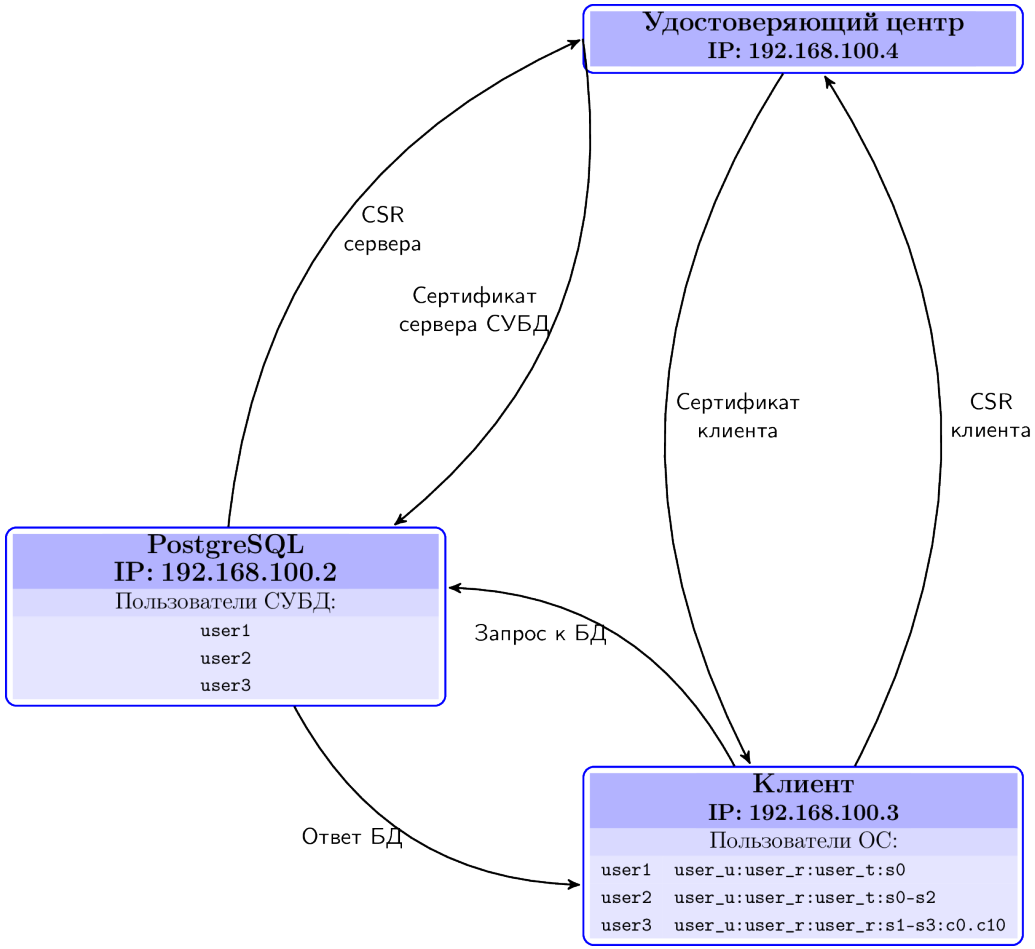
Механизм многоэкземплярности:



Реализация: модуль `pam_namespace.so`
Скрипт инициализации: `namespace.init`
Конфигурационный файл: `namespace.conf`

Доработан модуль `pam_namespace`. Стало возможным передать значение текущего контекста пользователя в скрипт инициализации `namespace.init`.

Применение механизма для аутентификации клиентов СУБД PostgreSQL



Применение механизма для аутентификации клиентов СУБД PostgreSQL

```
                52:5f
            Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Selinux Context:
                user_u:user_r:user_t:s0-s2
            X509v3 Basic Constraints: critical
                CA:FALSE
    Signature Algorithm: sha1WithRSAEncryption
         42:85:e6:16:ef:5a:bf:b4:a7:87:44:0d:50:3f:2b:79:2e:87:
         a3:8f:cc:da:0d:0d:85:62:b1:a7:53:7b:bd:15:86:f6:cd:3f:
         b7:38:0d:91:d6:8b:a4:58:45:66:49:45:3c:13:00:b9:1b:e7:
         3c:44:6b:51:cf:7e:6c:1f:fc:01:03:84:6d:42:7a:96:26:33:
         e5:44:33:cd:c2:55:63:9e:92:21:f3:31:75:7b:da:38:bd:e7:
         2f:fe:ab:c1:a8:55:ea:d1:12:2f:aa:0f:fe:63:c4:a0:c0:b7:
         ef:f7:8b:49:83:60:78:34:dc:50:48:31:00:21:e4:09:ed:38:
         96:9f:6d:5f:9f:a9:6a:83:86:5f:a1:ec:be:4d:d3:99:f0:80:
         f2:e3:47:36:47:f4:c7:b6:d9:4b:0d:be:e5:9c:7e:85:7a:ec:
         4f:ad:46:e9:21:38:08:84:71:be:3b:fd:ab:0a:42:d6:e7:d5:
         14:8b:9d:9c:b5:45:78:22:39:bb:3e:52:da:1b:c3:3b:10:5c:
         40:9c:4d:50:6d:ee:1d:f3:4c:3a:be:ee:47:53:54:d6:b9:e5:
         e3:67:ac:c4:75:87:55:39:65:23:88:68:cd:b8:9f:e6:bd:c4:
         e8:af:88:11:d1:fd:d3:f0:47:d7:71:3c:df:b0:00:5a:ac:ec:
         91:ea:1a:f2

[user2@client ~]$ _
```

Применение механизма для аутентификации клиентов СУБД PostgreSQL

```
     ¦ обычная
 public ¦ ssl_get_extension_by_name ¦ text                   ¦ text
     ¦ обычная
 public ¦ ssl_get_extensions_count  ¦ text                   ¦
     ¦ обычная
 public ¦ ssl_is_critical_extension ¦ text                   ¦ text
     ¦ обычная
 public ¦ ssl_is_used               ¦ boolean                ¦
     ¦ обычная
 public ¦ ssl_issuer_dn             ¦ text                   ¦
     ¦ обычная
 public ¦ ssl_issuer_field          ¦ text                   ¦ text
testdb=> SELECT ssl_get_extension_by_name('selinuxContext');
 ssl_get_extension_by_name
---------------------------
 user_u:user_r:user_t:s0-s2
(1 строка)

testdb=> SELECT sepgsql_getcon();
       sepgsql_getcon
---------------------------
 user_u:user_r:user_t:s0-s2
(1 строка)

testdb=> _
```

Заключение

- 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.

Спасибо за внимание!