Разработка виртуального HSM для платформы Linux

Карташов Д. А., Савенко С. А.

Parallels Lab SPbAU

Введение

Криптография в приложениях:

- хранение секретных данных
- вычисления с их использованием

Проблемы безопасности:

- компрометация секретных данных

Решение:

- исключить попадание секретных данных на диск и/или в память компьютера

Цели и задачи проекта

Цель

Разработать решение, предоставляющее функциональность HSM в виртуальном окружении

Задачи

- поиск и анализ существующих решений;
- изучение стандартов HSM;
- изучение приложений, поддерживающих HSM;
- разработка прототипа VHSM.

Существующие решения

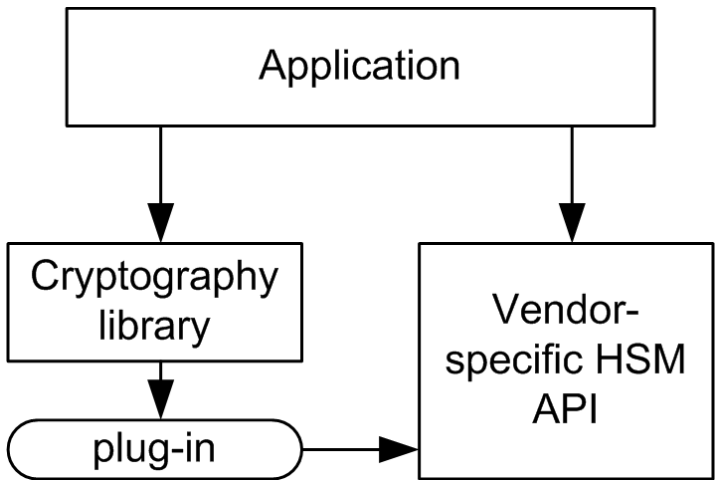
Существующие решения:

- Amazon CloudHSM:
  `http://aws.amazon.com/cloudhsm/`

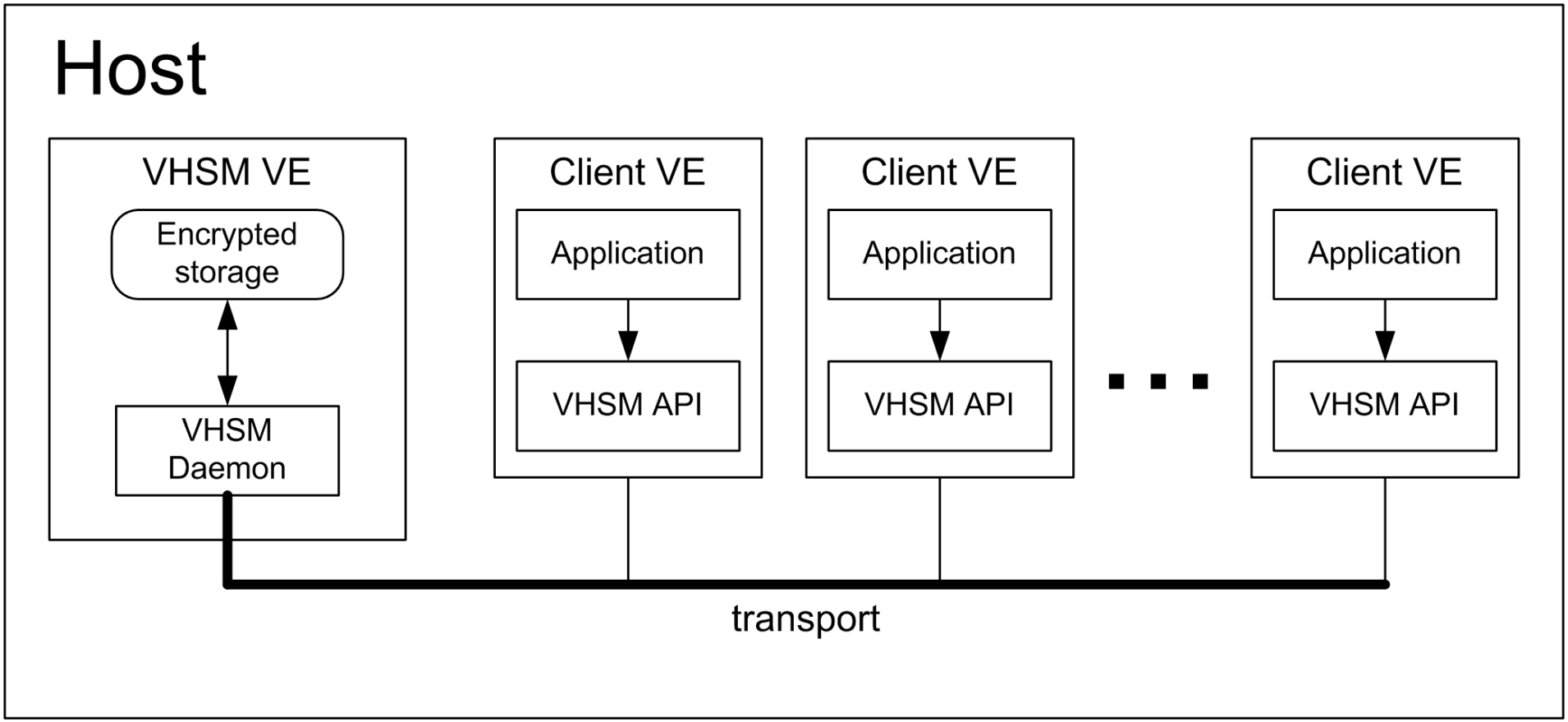
Стандарты HSM:

- pkcs#11:
  `http://www.rsa.com/rsalabs/node.asp?id=2133`

Использование HSM в приложениях:



Общая архитектура решения



Основные компоненты

- **Transport**
  - пересылка сообщений
  - идентификация контейнеров
- **VHSM API**
  - передача запросов на выполнение криптографических операций через транспорт
  - получение результатов операций
- **VHSM daemon**
  - аутентификация
  - выполнение криптографических операций с использованием секретных данных
- **Encrypted storage**
  - хранение секретных данных пользователя

Transport

- протокол — Google Protobuf
- реализован на файловой системе
  - структура каталогов:

```
transport
|__ containers
|    |__ CNT1
|    |    |__ in
|    |    |__ out
|    |__ CNT2
|         |__ in
|         |__ out
|__ vhsm
     |__ in
     |__ out
```

VHSM API

- управление сессиями

```
vhsm_rv vhsm_login(vhsm_session session,
                   vhsm_credentials credentials);
vhsm_rv vhsm_logout(vhsm_session session);
```

- управление ключами

```
vhsm_rv vhsm_key_mgmt_delete_key(vhsm_session session,
                                 vhsm_key_id key_id);
vhsm_rv vhsm_key_mgmt_create_key(vhsm_session session,
                                 vhsm_key key);
```

- хэширование и MAC

```
vhsm_rv vhsm_mac_init(vhsm_session session,
                      vhsm_mac_method method);
vhsm_rv vhsm_mac_update(vhsm_session session,
                        unsigned char const * data_chunk,
                        unsigned int chunk_size);
vhsm_rv vhsm_mac_end(vhsm_session session,
                     unsigned char * mac_ptr,
                     unsigned int * mac_size_ptr);
```

VHSM daemon и encrypted storage

- вычисление криптографических функций
- хранение пользовательских данных
  - генерация ключа для шифрования с использованием секрета пользователя
  - шифрование AES в режиме GCM
  - реализовано на файловой системе

```
encrypted_storage
|__ .fses_root
|__ user1
|    |__ .fses_ns
|    |    |__ .fses_ns_cf
|    |__ key1
|__ user2
     |__ .fses_ns
     |    |__ .fses_ns_cf
     |__ key1
```

OpenSSL engine

OpenSSL engine — модуль расширения для OpenSSL, используемый для делегирования криптографических функций VHSM.

- динамический;
- статический (встроен в `libcrypto`).

Различия при использовании стандартных и нестандартных алгоритмов хэширования и шифрования:

- нестандартные алгоритмы могут использоваться только через унифицированные интерфейсы;
- функциональность стандартных алгоритмов может быть перегружена по запросу пользователя.

OpenSSL engine — детали реализации

В текущей реализации движка изменен алгоритм SHA1 так, чтобы он смог разграничить фазы вычисления HMAC и передавать в VHSM корректные данные.

Минусы:

- алгоритм работы движка основан на текущей реализации функции HMAC в OpenSSL, и при ее изменении он станет некорректным;
- алгоритм SHA1 ведет себя не так, как этого может ожидать пользователь.

Плюсы:

- от конечного пользователя не потребуется значительных усилий по внедрению поддержки VHSM в свое приложение.

OpenSSL engine — конфигурация

В файл `/etc/ssl/openssl.cnf` необходимо добавить следующие строки:

- в начале файла:

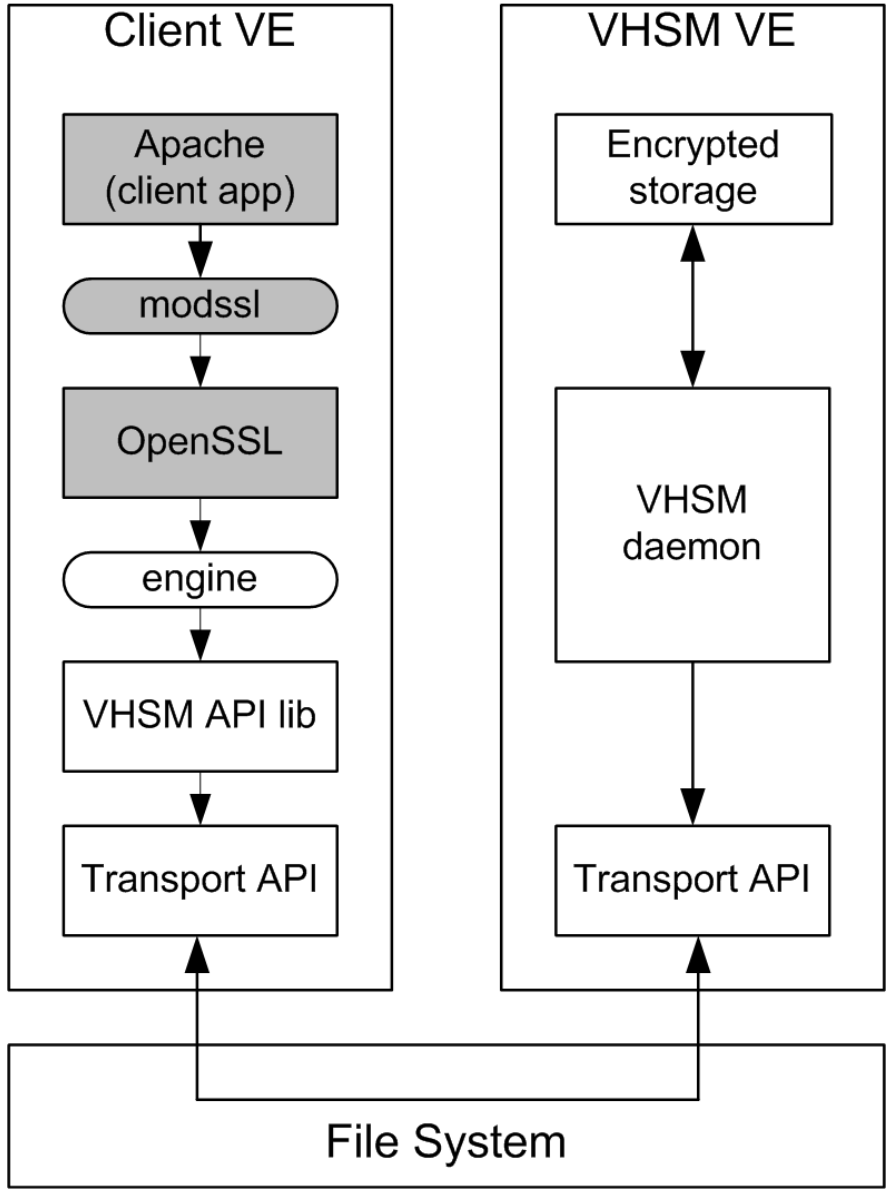
```
openssl_conf = openssl_def
```

- в конце файла:

```
[openssl_def]
engines = engine_section
[engine_section]
test_engine = test_engine_section

[test_engine_section]
engine_id = test_engine
dynamic_path = /path/to/test_engine.so
username = user
password = password
init = 0
```

Пример использования



Итоги и планы

Реализован прототип:

- MAC
- хэш-функции
- управление ключами
- OpenSSL engine

Планы на будущее:

- Повышение надёжности шифрования
- Реализация транспорта через netlink / pipe
- Расширение функциональности
- Реализация системы конфигурации
- Тестирование
- Рефакторинг

Ссылки

- Репозиторий проекта:
  `https://github.com/OSLL/vhsm`
- wiki проекта:
  `http://osll.spb.ru/projects/vhsm/wiki`