1. Охарактеризовать статический и динамический методы анализа программ.

Статический метод анализа программ

При статическом методе защищаемая программа сначала дизассемблируется. По полученному ассемблерному коду локализуются механизмы защиты, изучается логика их работы.
Наиболее распространенная программа дизассемблирования - **IDA (interactive disassembler)**.

Динамический метод анализа программ

При динамической нейтрализации изучение и реконструирование логики работы защитных механизмов производятся с помощью отладчиков или эмуляторов, а необходимые изменения вносятся непосредственно в код программы.
Отладчик: **SoftIce**

2. Охарактеризовать модели аппаратных ключей HASP HL

Модель ключа HASP HL	Размер памяти	Количество лицензий	Основные характеристики
Basic	нет	1	шифрование/ дешифрование
Pro	112 байт	16	шифрование/ дешифрование, Hasp HL ID
Max	4096 байт	112	шифрование/ дешифрование, Hasp HL ID
Time	4096 байт	112*	шифрование/ дешифрование, Hasp HL ID, Часы реального времени
Net 10	4096 байт	112	шифрование/ дешифрование,

Модель ключа HASP HL	Размер памяти	Количество лицензий	Основные характеристики
			Hasp HL ID, Сетевой доступ
Net 50	4096 байт	112	шифрование/ дешифрование, Hasp HL ID, Сетевой доступ
Net 250	4096 байт	112	шифрование/ дешифрование, Hasp HL ID, Сетевой доступ

Все ключи HASP простроены на ASIC-чипах

HASP HL Basic

- Аппаратно реализованный алгоритм шифрования AES/128;
- кроссплатформенность.

HASP HL Pro

- защита до 38 приложений или модулей на 1 ключ;
- уникальный ID;
- AES/128;
- защищенная память;
- возможность дистанционного перепрограммирования ключа.

HASP HL Max

- Защита до 233 локальных приложений или модулей на 1 ключ;
- защищенная память;
- уникальный ID;
- AES/128;
- возможность дистанционного перепрограммирования ключа.

HASP HL Net

- защита сетевых приложений;

- ограничение на количество одновременно работающих пользователей;
- защита до 233 сетевых приложений или модулей на 1 ключ;
- защищенная память;
- уникальный ID;
- AES/128;
- возможность дистанционного перепрограммирования ключа.

Hasp HL Time

- организация аренды, лизинга, подписки на обновления;
- распространение trial версий;
- защита до 233 локальных приложений или модулей на 1 ключ;
- защищенная память;
- уникальный ID;
- AES/128;
- возможность дистанционного перепрограммирования ключа.

3. Охарактеризовать защиту на основе HASP SRM Envelope

Пристыковочный способ защиты

Принцип **конверта** - у нас есть *незащищенное приложение*. При помощи HASP SRM Envelope мы его *упаковываем в конверт*, который делает его безопасным.

Защита с помощью HASP SRM Envelope

Можно:

1. Защищать приложения
2. Защищать файлы данных WIN32

Чтобы защитить приложение:

1. Запустить утилиту HASP SRM Envelope
2. ввести путь до защищаемого приложения
3. ввести пароли HASP
4. нажать Protect

Чтобы защитить файлы данных

1. Во вкладке Envelop
    - ввести путь
    - пароли
    - собственный Encryption Key
2. Выполнить защиту данных.

4. Охарактеризовать защиту с помощью HASP API.

Встроенный способ защиты

1. Проанализировать код приложения, в том числе на наличие ошибок.
2. Встроить в приложение код вызовов к HASP API
3. Скомпилировать.

Группы сервисов HASP API:

- HASP Login
    - Устанавливает сеанс доступа к Компоненту и определяет контекст сессии.
- HASP Get Info
    - Применяется для запроса информации о системных компонентах в соответствии с заданными параметрами поиска.
- HASP Encrypt
- HASP Decrypt

5. Провести обзор функций HaspSRM Run-timeAPI

Сессия

hasp_login()

```
Устанавливает сеанс доступа к Компоненту и определяет контекст сессии.
```

hasp_logout()

```
Завершение текущего сеанса
```

hasp_get_session_info()

Получение данных об условиях сеанса

Время

hasp_datetime_to_hasptime()

Преобразование значений даты и времени в hasptime

hasp_hasptime_to_datetime()

Преобразование значения времени с hasptime в формат даты и времени.

hasp_get_rtc()

Считывание текущего времени из ключа HASP HL Time или HASP HL NetTime

Криптография

hasp_decrypt()

Расшифрование данных из буфера с использованием AES

hasp_encrypt()

Шифрование данных в буфере с использованием AES

Файлы

hasp_read()

Считывание данных на ключе HASP SRM

hasp_update()

```
Запись обновленной лицензии на ключ HASP SRM
```

hasp_write()

```
Запись данных в память ключа HASP SRM
```

hasp_get_size()

```
Извлечение данных об объеме файла в памяти
```

hasp_free()

```
Освобождение ресурсов из памяти
```

Другое

hasp_get_info()

```
Получение данных в соответствии
с заданными параметрами поиска
и представление их в заданном формате
```

6. Провести сравнение Hasp SRM Envelop и Hasp SRM Run-time API

Hasp SRM Envelope	Hasp SRM Run-time API
Оперативная защита приложения в автоматическом режиме	Все обращения к API-функциям необходимо прописывать в исходном коде вручную
Определение особых параметров защиты ПО	Контролируемый и кропотливый процесс, обеспечивающий максимальную защиту. Степеть защищенности зависит от объема функций API, реализованных при создании защиты

Hasp SRM Envelope	Hasp SRM Run-time API
Отсутствует необходимость редактирования исходного кода	Исходный код необходимо редактировать вручную
Антиотладочные механизмы и защита от обратного инжиниринга	Максимальная гибкость

7. Основные стратегии защиты программ с помощью HASP API при использовании ключа Hasp.

Использование множественных вызовов

- Вставляйте в программный код приложения множественные вызовы процедуры `hasp()`, что способно существенно затруднить попытки взлома. Комплексные множественные вызовы значительно усложняют отслеживание и взлом схемы защиты.

Шифрование внешних и внутренних данных

- в качестве объекта шифрования можно выбрать заголовки файлов, важные для расчетов, постоянные или небольшие поля БД.
- все, что способно повлиять на основные функциональные возможности приложения, может являться потенциальным объектом шифрования.
- необходимо предусмотреть **извещение пользователя о том, что соответствующий ключ HASP не подключен**

Отсутствие повторяющихся схем защиты

Недостатки повторяющихся схем

1. Повторяющаяся в программном коде схема легко отслеживается. После того, как вашу схему распознали, её будет легко найти и взломать в любой части программного кода.
2. Если приложение содержит большое количество исполняемых файлов и библиотек, следует использовать разные схемы защиты для этих файлов.

Разделение шагов вызовов

- Проверка HASP подразумевает выполнение трех шагов:
  1. вызов процедуры `hasp()`
  2. оценка значений, полученных от защитной процедуры
  3. ответ в зависимости от полученных значений
- для повышения безопасности следует поместить каждый из этих шагов в **различные места**.

Шифрование памяти аппаратного ключа

8. Использование функций шифрования и расшифрования для защиты программ с помощью ключа HASP HL.

- в качестве объекта шифрования можно выбрать заголовки файлов, важные для расчетов, постоянные или небольшие поля БД.
- все, что способно повлиять на основные функциональные возможности приложения, может являться потенциальным объектом шифрования.
- необходимо предусмотреть **извещение пользователя о том, что соответствующий ключ HASP не подключен**

Встроенный криптопроцессор ключей HASP SRM осуществляет криптографические операции на основе алгоритма AES.
При шифровании HASP SRM использует набор секретных 128-бит ключей,
которые не покидают памяти ключа HASP SRM

9. Привести основные сведения о типичной микропроцессорной карточке.

Типичные характеристики микропроцессорных пластиковых карт:

1. Криптостойкость: DES, 3DES, AES
2. Защита внутренних ресурсов при помощи одного или нескольких секретных кодов.
3. Объект EEPROM: от 64 байт до десятков килобайт

```
EEPROM - Ellectrically Erasable Programmable Read-Only Memory
Электрически стираемое перепрограммируемое ПЗУ
```

4. Типы поддерживаемых протоколов связи: T=0, T=1
5. Поддерживаемые стандарты:
    - ISO 7816
    - GSM
    - EMV
    - Java Card
6. Напряжения питания: в зависимости от класса карты:
    - 1.8 В
    - 3.3 В
    - 5.0 В
7. Потребляемый ток: по 15 мА.

Сферы применения

1. Идентификация личности
2. Контроль доступа к вычислительным ресурсам
3. Разграничение доступа к закрытым и охраняемым помещениям
4. Верификация электронной подписи.
5. Реализация алгоритмов шифрования
6. Генерация паролей
7. Хранение защищенных данных пользователя или организации.

10. Охарактеризовать методы защиты смарт-карт от подделки.

1. Эмбоссирование;
    - Нанесение данных на карточке в виде рельефных знаков
2. Применение специальных объемных голографических изображений;
3. Фотография владельца карты;
4. Специальная полоска для подписи владельца;
5. Использование специальных красителей.

11. Структуры и типы команд протокола T=0 стандарта ISO 7816-3 для интеллектуальных карт.

Протокол типа T=0

- Передача символов
- Всегда однонаправленный
- Осуществляется с помощью команд, структура ниже

class	INStruction	P1	P2	P3	Optional Data

```
P1, P2 - параметры определенной команды,
Если P3=0, то предполагается, что длина передаваемой информации 256 байт.
Optional Data - структура для команды записи данных ниже
```

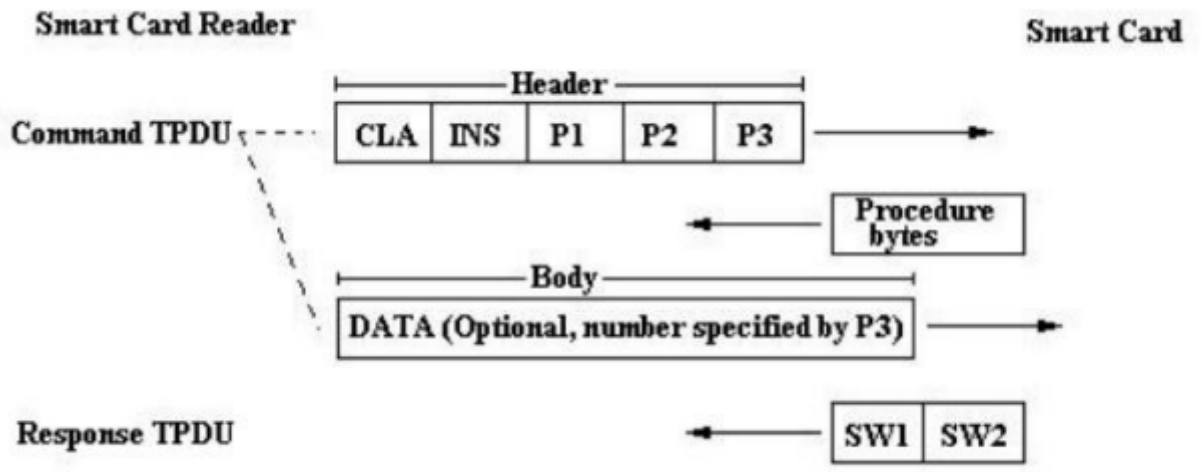
Optional Data	SW1	SW2

```
SW1, SW2 - статус выполняемых команд.
Если SW1 = 90h, SW2 = 00h, то успешно завершение.
```

Если осуществляется команда чтения данных, то Optional Data предназначается для считывания данных

Схемы из презентации

Code	Description
CLA	Class of command
INS	INStruction code
P1	Instruction parameters
P2	Instruction parameters
P3	Indicates the number of bytes to be transfered during the command



12. Структуры передачи информации в протоколе T=1 стандарта ISO 7816-3.

Протокол типа T=1

- **Блочная передача данных**
  - блоки символов помещаются в специальную оболочку
- **Операции**
  1. Контроль передачи;
  2. Сцепление блоков символов;
  3. Коррекция ошибок.
- **В терминах Приложений данные определяют операции:**
  1. Работа с файлами;
  2. Команды аутентификации;
  3. Команды изменения прав доступа;
  4. Команды криптографических преобразований.

Поля протокола передачи

NAD	PCB	LEN	INF	EDC
указывается адрес источника и приемника	для контроля	длина	информация	блок обнаружения ошибок
1 байт	1 байт	1 байт	от 0 до 254 байт	1 или 2 байта

NAD

Node Address Destination
Это адрес точки назначения.
Под поле отводится 1 байт, который состоит из двух полубайт:

1. источник (*приложение*)
2. приемник (*карта*)

PCB

Protocol Control Byte
Это контрольный байт протокола.
Содержит информацию, необходимую для контроля передачи.

Виды PCB

1. **I-block**
    - Информационный блок
    - Передача информации
    - Пересылка идет поблочно
2. **R-block**
    - Блок готовности к приему
    - Сигнализирует о том, что пойдет пересылка цепочки блоков
3. **S-block**
    - Блок контроля
    - Для обмена контрольной информацией между считываетелм и карточкой
    - информация о правильности приёма блоков

LEN

Число байт, которые находятся в поле `INF`

INF

Информационное поле.
Содержит данные команды, которые необходимо переслать карточке или данные, получаемые от карточки.

EDC

Error Detection Code
Блок обнаружения и коррекции ошибок.

13. Структуры и типы команд по стандарту ISO 7816-4.

В прикладных программах реализованы функции для работы с конкретными типами карт или с клиентскими приложениями на картах.

Простая прикладная программа может обеспечивать управление обменом блоками APDU (Application Protocol Data Unit) с картой.

Протокол ISO 7816-4: APDU команды

Связан с прикладным уровнем передачи информации.
Заголовок описывает класс команды и тип передаваемой инструкции.
Заголовок запроса

CLA	INS	Le	P1	P2	DATA	Lc

```
> P1 и P2 - Характеризуют конкретный тип команды
> Data - Данные, до 255 байт
> c - длина передаваемых на карточку данных
> e - количество считываемых байтов с карточки
```

Заголовок ответа

Data	SW1	SW2

```
> SW1 - статус
> SW2 - ситуация
> Data - данные.
    >> Считываются с карточки
    >> определяется Le
```

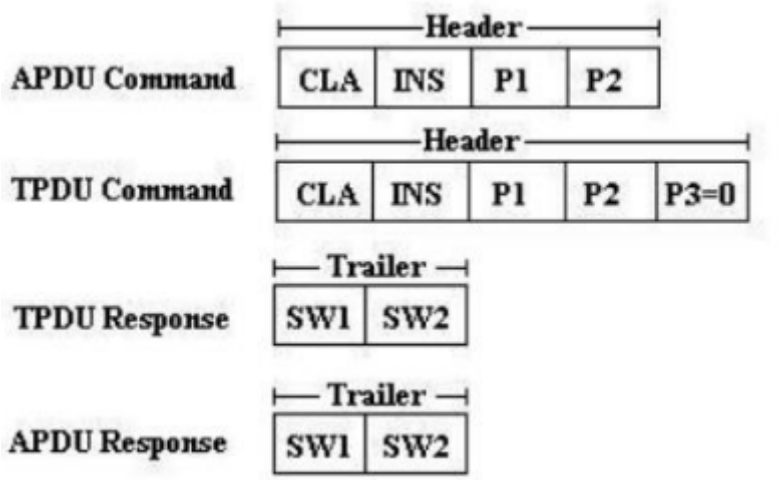
Команды позволяют

1. Стереть байты из файла на карточке;
2. Выбрать файл;

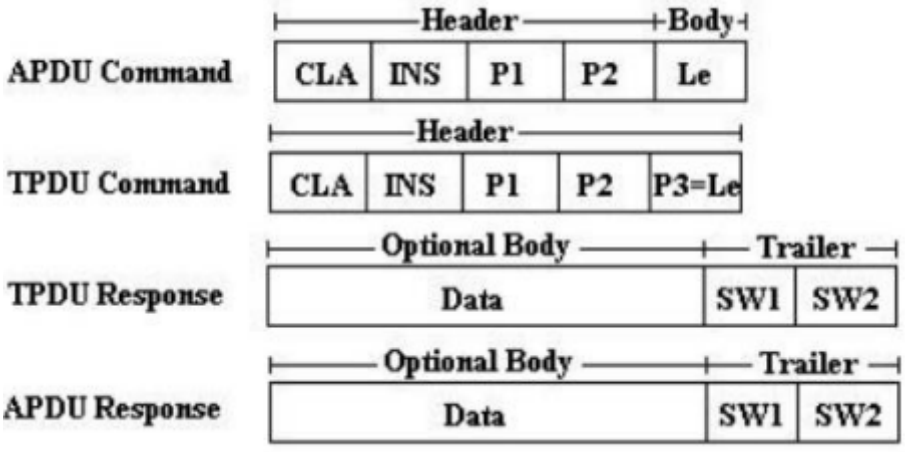
3. Получить ответ;
4. Получить данные;
5. Записать байты в текущий файл;
6. Получить от карты случайное число (важно дла аутентификации);
7. Внешняя аутентификация;
8. Внутренняя аутентификация.

Usage Cases

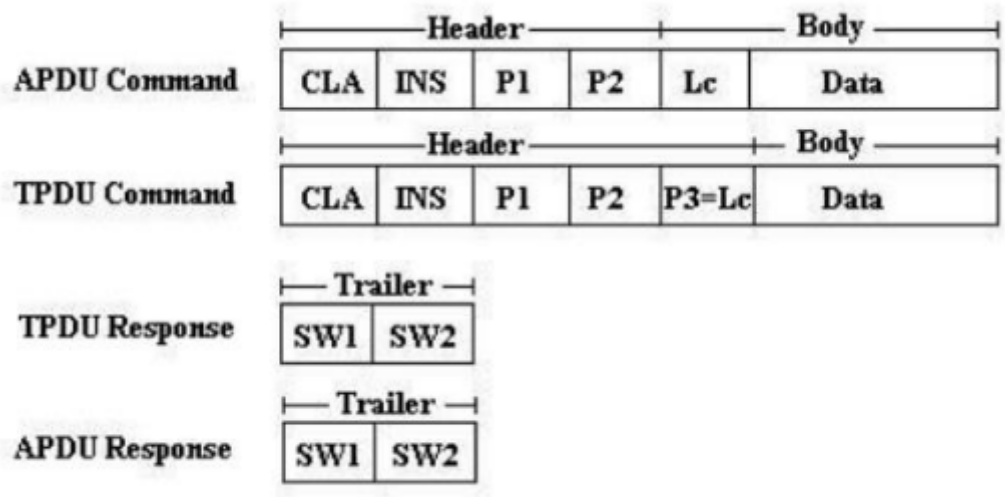
Передача данных отсутствует



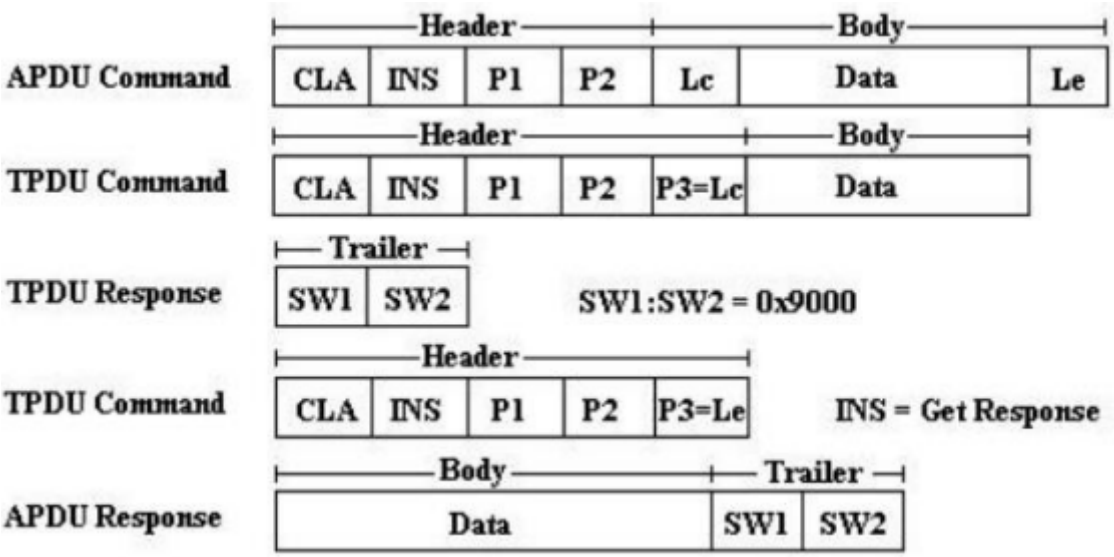
Передача данных от карты приложению



Передача данных на карту приложением



Передача данных на карту и считывание с карты приложением



14. Назначение смарт-карт eTokenPro.

eToken - персональное средство аутентификации и хранения данных, аппаратно поддерживающее работу с цифровыми сертификатами и ЭЦП.
Выпускается в форм-факторах USB-ключа или смарт-карты.

Назначение eTokenPro

- двухфакторная *аутентификация* пользователей при доступе к защищенным ресурсам по сертификатам X.509, хранимым в памяти ключа и одноразовым паролям.
- аппаратное выполнение криптографических операций в доверенной среде.
- безопасное хранение:
  - криптографических ключей
  - профилей пользователей
  - вычисление ХЭЩ-функции
  - формирование ЭЦП

15. Основные характеристики eToken Pro.

Характеристика	eToken PRO/16K	eToken PRO/32K	eToken PRO/64K
Тип смарт-карты	Infineon SLE66CX160S	Infineon SLE66CX320P	Infineon SLE66CX640P
Размер EEPROM	16 КБ	32 КБ	64 КБ
Размер ROM	64 КБ		136 КБ
Размер RAM	3004 байт		5052 байт
Частота процессора	6 МГц		
Время хранения информации	Не менее 10 лет		
Количество циклов перезаписи одной ячейки памяти EEPROM	Не менее 500000 раз		
Операционная система	Siemens CardOS/M4	Siemens CardOS/M4.01	Siemens CardOS/M4.2
Файловая система	Siemens CardOS/M4		
Криптографические возможности	Асимметричные криптографические алгоритмы: RSA, DSA. Симметричные алгоритмы шифрования: DES, 3DES. Хэш-функции: SHA-1. Дополнительно: Аппаратный генератор случайных чисел (ГСЧ).		
Подпись RSA/1024 результата операции хэширования (128 бит)	Не более 1 с		
Генерация RSA ключа (1024 бит)	Не более 15 с		

16. Четыре области памяти в смарт-картах eTokenPro

Системная

Содержит файловую и операционную системы. В ней хранятся имя eToken и данные, необходимые для проверки правильности вводимых PIN-кодов и паролей администратора.

Закрытая

Содержит данные, для доступа к которым требуется ввод PIN-кода или пароля администратора.

Открытая

Данные, для доступа к которым требуется только подключение eToken.

Свободная

Не содержит никаких данных.

```
Открытая, закрытая и свободная области памяти не имеют фиксированных границ.
Возможность доступа к той или иной области
памяти электронного ключа eToken PRO зависит от наличия необходимых прав.
```

17. Основные факторы, определяющие безопасность eToken PRO

- **Операционная система**.
  - Устойчивость к взлому операционной системы процессорной смарт-карты (включая механизмы идентификации, аутентификации, контроля доступа, обмена данными), подтверждённая международной сертификацией ITSEC.
- **Микросхема смарт-карты**.
  - Защищённость микросхемы смарт-карты от физического считывания содержимого внутренней памяти EEPROM.
- **Логическая организация данных**.
  - Дополнительная защита секретных данных от считывания благодаря шифрованию при хранении.
- **Корпус USB-ключа**.
  - Конструкция, не поддающаяся не обнаруживаемому вскрытию корпуса.
- **Защита трафика от USB-прослушивания**.

- Защита передаваемых данных с помощью механизма Secure Messaging.

18. Уровни доступа к информации в смарт-картах eTokenPro.

Гостевой

1. Возможность просматривать объекты в открытой области памяти;
2. Возможность получения из системной области памяти общей информации относительно eTokenPRO.

Чтобы пользоваться, достаточно подключить eToken PRO.
Пример использования: чтение уникального идентификатора eToken.

Пользовательский

1. Право просматривать, изменять и удалять объекты:
   - в открытой области памяти;
   - защищенные PIN-кодом в закрытой области памяти.
2. Возможность получения общей информации относительно электронного ключа eToken PRO;
3. Право выполнять криптографические операции;
4. Право менять PIN-код.

Чтобы пользоваться, необходимо ввести PIN-код.

Административный

1. Право просматривать, изменять и удалять защищенные паролем администратора объекты в закрытой области памяти.
2. Право менять PIN-код, не зная его;
3. Право смены пароля администратора;
4. Право настраивать параметры безопасности eToken PRO
5. Право делать параметры безопасности доступными в пользовательском режиме.

Чтобы пользоваться нужно ввести пароль администратора.

Смешанный доступ

Сочетание административного и пользовательского доступа.

Доступ Эмитента

1. Право переформатирования ключа.
2. Установка нового пароля администратора.
3. PIN-кода пользователя.

Чтобы пользоваться необходимо ввести ключ форматирования в процессе форматирования eToken.

19. Архитектура программного обеспечения, определяемая стандартом PC/SC.

PC/SC - (*Personal Computer / Smart Card*) - набор спецификация для доступа к смарт картам.

Стандарт PS/SC определяет состав архитектуры, протоколов, компонентов и программных интерфейсов, которые должны предоставлять производители в своих продуктах для обеспечения взаимодействия между смарт-картами.

Компоненты архитектуры PS/SC

ICC (Integrated Circuitry Card)

```
Смарт-карта
```

IFD (Interface Device)

```
> Устройство для связи со смарт-картой
> Устройство чтения смарт-карт
```

IFD Handler

```
> Программное обеспечения для IFD
> Драйвер считывателя
```

Resource Manager

```
> Системный компонент, управляющий ресурсами, связанными со смарт-картами
```

Приложения, поддерживающие ICC

```
> приложения, которые используют смарт-карты
```

Поставщик услуг

```
> Программная библиотека, используемая приложениями для работы со смарт-картой
```

20. Охарактеризовать среду программирования eTokenRTE (RunTimeEnvironment).

eToken RTE

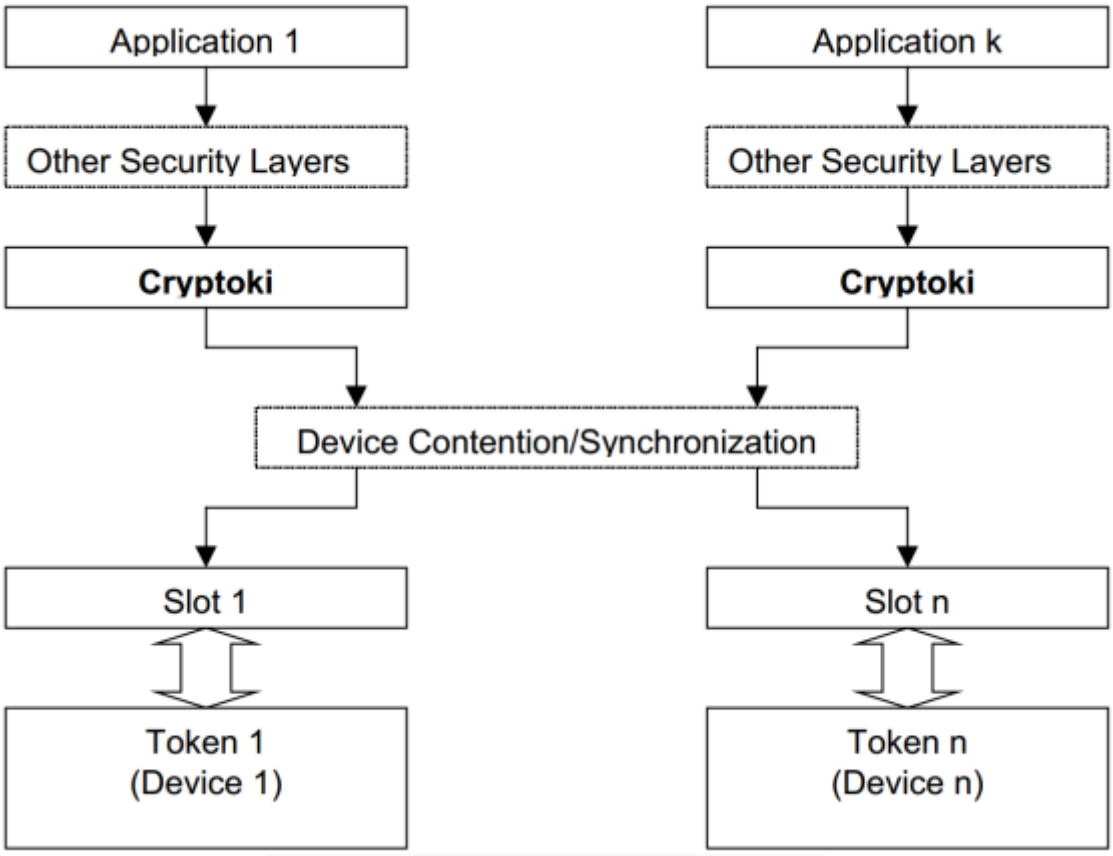
- набор необходимых **программных компонентов** для поддержки eToken в различных операционных системах и приложениях.
- должен быть **установлен** на каждом компьютер, где **будет использоваться eToken**
- предназначен для обеспечения **поддержки всех моделей eToken в ОС Windows**
- для поддержки функционирования **в других ОС, предлагает отдельные программные решения.**

21. Интерфейсы, обеспечиваемые в eTokenRTE

PKCS#11

```
Public-Key Cryptography Standards
```

1. Стандарт криптографии с открытым ключом
2. Опубликован RSA Laboratories.
3. Определяет платформо-независимый программный интерфейс доступа к криптографическим устройствам.
4. Другое название **Cryptoki** (*cryptographic token interface*)
5. Alladin использует его как главный API для программирования eToken



Microsoft CryptoAPI

1. ОС Windows использует.
2. Применяется как пользовательскими приложениями, так и самой системой.
3. Две основные ветви функциональности
    - механизм шифрования и хранилище ключевых контейнеров
    - работа с цифровыми сертификатами X.509

Для поддержки eToken в рамках CryptoAPI используются как единый компонент:

- поставщик средств криптографии (CSP) для eToken - *eToken Base Cryptographic Provider*
- поставщик хранилища сертификатов eTCertStore для подключения физического хранилища сертификатов на eToken.

SAPI

```
Supplementary API
```

1. Впервые применен в eToken RTE 3.60 с целью внедрения в приложения низкоуровневых API.
2. Предоставляет доступ к специфическим функциям eToken, не охваченным стандартом PKCS#11.

3. Поддерживается в PKI Client

22. Охарактеризовать RFID – системы

RFID - *Radio Frequency Identification* - радиочастотная идентификация

- Способ автоматической идентификации объектов, в котором посредством радиосигналов *считываются или записываются данные, хранящиеся в* ***траснпондерах*** *или RFID-метках*

Составные части

1. Считыватель (ридер, интеррогатор)
2. Транспондер (RFID-метка, RFID-тег)
    - (*transmitter-responder*)
    - приемопередающее устройство, посылающее сигнал в ответ на принятый.

По дальности считывания

1. Системы ближней идентификации (до 20 см)
2. средней дальности (20см до 5м)
3. дальней идентификации (5 до 300м)

Части RFID-меток

1. Интегральная схема для
    - хранения и обработки информации,
    - модулирования и демодулирования радиочастотного сигнала
2. Антенна
    - для приема и передачи сигнала

23. Бесконтактные карты. Принцип работы

Бесконтактная карта - общее название бесконтактных устройств на интегральных схемах, используемых для разграничения доступа или в платежных системах.

- Старые *125 кГц*
- Новые *13,56 МГц* RFID

Стандарты:

- ISO 14443 (*proximity card*)
- ISO 16693 (*vicinity card*)

В основе - *колебательный контур*:

- конденсатор
- катушка индуктивности

1. Устройство чтения излучает переменное электромагнитное поле стандартной частоты, что возбуждает переменный электрический ток в катушке индуктивности и в колебательном контуре.
2. Этот ток преобразуется в постоянный и заряжает конденсатор, который питает электроэнергией микросхему.
3. **Обмен** информацией между картой и считывающим устройством осуществляется **через ту же катушку посредством модуляции** колебаний электромагнитного **поля устройства**

В простейших случаях карта циклически передает свой ID.
В более сложных моделях происходит двусторонний обмен информацией по принципу запрос-ответ.

NFC

```
Near Field Communication
```

Позволяет поддерживающим её мобильным устройствам (**телефонам и планшетам**)

- выступать в роли виртуальной бесконтактной карты
- работать в качестве устройства чтения по стандарту ISO 14443

работает в малом радиусе действия.

24. Бесконтактные карты MIFARE

MIFARE - торговая марка семейства бесконтактных смарт-карт.

- Считается наиболее распространенной торговой маркой бесконтактных смарт-карт в мире
- Все продукты базируются на ISO 14443 Type A `13.56 МГц`
- Предназначены для:

    - идентификации личности
    - микроплатежных систем
- Характеризуются невысокой дальностью чтения до `10 см`

Mifare Classic

- Объем карты составляет `1 кб` или `4 кб`
- стандарт EEPROM
- батарея питания не требуется
- надежно разграниченные между собой 16 ил 40 секторов, поддерживающие многофункциональное применение.
- каждый сектор имеет свой набор ключей доступа, что позволяет разграничивать доступ к различным приложениям.
- каждый сектор состоит из 4 блоков
    - 3 информационных
    - 1 для хранения ключей
- блок - самый малый компонент, к которому адресуется пользователь, состоит из 16 байт
- срок хранения данных в памяти до 10 лет
- до 100_000 циклов перезаписи
- Время, требующееся для получения идентификаторов карты - `3 мс`
    - старт
    - ответ на запрос
    - антиколлизия
    - выбор
- Время считывания 16-байт блока
    - `2.5 мс` (без аутентификации)
    - `4.5 мс` (с аутентификацией)
- Типичная операция по выдаче билета `< 100 мс`
- проведение операций возможно, когда карта находится в движении

Криптоалгоритм Crypto-1

- изначально была сокрыта реализация, криптостойкость была основана на его секретности
- на самом деле криптостойкость - низкая, на сегодняшний день он не секретен и легко взламывается.

Проблемы с безопасностью

Все современные микросхемы считывателей Mifare умеют работать с Crypto-1.
Но не все имеют возможность безопасного энергонезависимого хранения ключей.

`В MFRC52x и NFC ключи подгружаются перед каждой транзакцией по незащищенному интерфейсу`

В остальных микросхемах ключ записывается **однократно** энергонезависимо и не может быть считан снаружи.

25. Бесконтактные карты EM-MARINE

Работают на системе `proximity` и соответствуют общемировым стандартам безопасности и качества.
Они дешевле, чем смарт-карты HID и Mifare.

Выпускаются на смарт-картах:

- Тонких (`0.8 мм`)
  - можно разместить магнитную полосу
  - можно разместить номер для идентификации
  - можно разместить полосы для подписи
  - чаще всего используются для СКУД
  - такой тип персонализируют при помощи:
    - термопечати
    - шелкографии
    - печати на офсете
- Толстых (`1.6 мм`)
  - такой тип персонализируют при помощи:
    - наклейки из пластика со всей нужной информацией

Рабочая частота	**125 КГц**
Тип микросхемы	TK4100 и TK28
Объём памяти	64 бит
Максимальное количество считываний	100 000 раз
Дальность считывания карты	стандартно 1-15 см
Тип памяти	только считывание
Размер карты	85,4 x 54,0 x 0,81 мм

Рабочая частота	125 КГц
Материал	ПВХ
Печать ID	да, при необходимости
Влажность	до 90%
Температура хранения	-40°C...+70°C
Совместимость с	EM4100, EM4102
Стандарт	ISO 18000-2

26. Охарактеризовать 2 схемы аутентификации (алгоритм, структура эталона), использующиеся в современных вычислительных системах.

Аутентификация - (*Authenication*)

- проверка принадлежности субъекта доступа по предъявленному им идентификатору
- подтверждение подлинности
- Может быть 2 типов:
  - односторонняя
  - взаимная

Схема 1.

```
Id_i = неизменный идентификатор i-го пользователя
K_i = аутентифицирующая информация пользователя,
      которая может изменяться и служит для аутентификации
      (например, пароль P_i = K_i)
```

Структура эталона

Номер пользователя	Идентификатор	Аутентификатор
1	Id_1	E_1 = F(Id_1, K_1)
2	Id_2	E_2 = F(Id_2, K_2)

Номер пользователя	Идентификатор	Аутентификатор
3	Id_3	E_3 = F(Id_3, K_3)
...		
n	Id_n	E_n = F(Id_n, K_n)

Протокол идентификации и аутентификации:

1. Предъявление Id
2. Проверка существования Id_i = Id для прохождения аутентификации
3. Запрос у пользователя его аутентификатора K.
4. Вычисление значения Y = F(Id_i, K).
5. Сравнение значений Y = E_i. Допуск при равенстве Y = E_i

- В данной схеме F - функция, которая обладает свойством "невосстановимости" значения K_i по E_i и Id_i
- невосстановимость оценивается пороговой трудоемкостью T_0 решения задачи восстановления K_i по E_i и Id_i.
  - На практике задают $T_0 = 10^{20} ... 10^{30}$
- Для пары K_i, K_m возможно совпадение соответствующих значений E.
- Вероятность ложной аутентификации не должна быть больше порогового значения P_0.
  - На практике задают $P_0 = 10^{-7} ... 10^{-9}$

Схема 2.

```
Id_i = неизменный идентификатор i-го пользователя
K_i  =  аутентифицирующая информация пользователя,
        которая может измениться и служит для аутентификации
        (например, пароль P_i = K_i)
S_i  = случайный вектор,
        создаваемый при создании идентификатора пользователя
```

Структура эталона

Номер пользователя	Идентификатор	Аутентификатор
1	Id_1, S_1	E_1 = F(S_1, K_1)
2	Id_2, S_2	E_2 = F(S_2, K_2)
3	Id_3, S_3	E_3 = F(S_3, K_3)

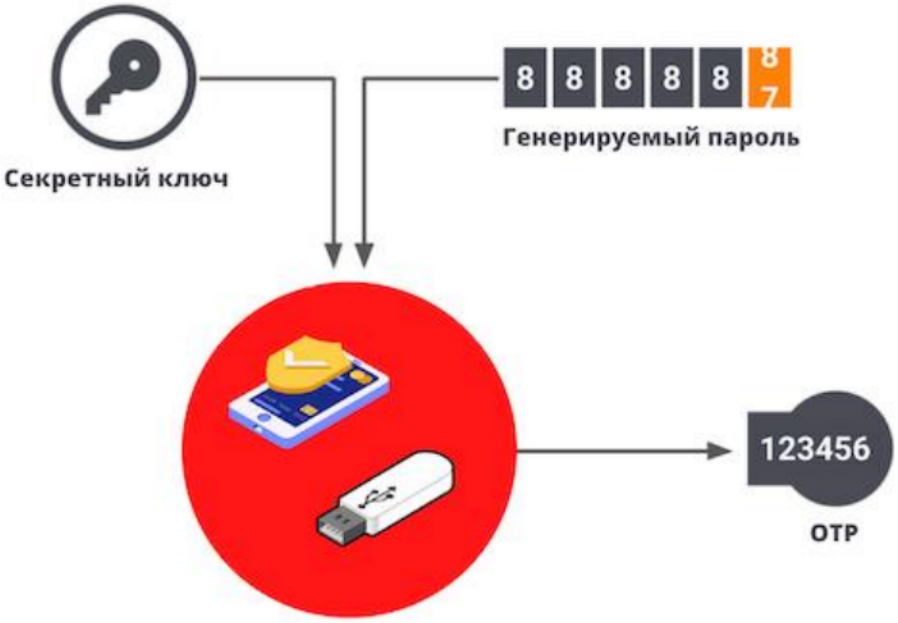
Номер пользователя	Идентификатор	Аутентификатор
...		
n	Id_n, S_n	E_n = F(S_n, K_n)

Протокол идентификации и аутентификации

1. Предъявление идентификатора Id
2. Проверка существования Id_i = Id для прохождения аутентификации
3. По идентификатору Id_i, выделяется вектор S_i
4. Запрос у пользователя его аутентификатора K
5. Вычисление значения Y = F(S_i, K)
6. Сравнение значений Y и E_i. Допуск при равенстве Y и E_i

27. Одноразовые пароли. Методы аутентификации с одноразовыми паролями.

One-time Password (OTP)



Технологии использования одноразовых паролей можно разделить на:

1. Использование генератора псевдослучайных чисел, единого для субъекта и системы.
2. Использование временных меток вместе с системой единого времени
3. Использование базы случайных паролей

Методы аутентификации с одноразовыми паролями

Использование генератора псевдослучайных чисел

Используется генератор псевдослучайных чисел с одинаков значение для субъекта и для системы.
Сгенерированный субъектом пароль может передаваться системе при последовательном использовании односторонней функции.

Использование временных меток вместе с системой единого времени

В качестве примера такой технологии можно привести SecurID от RSA.
Она основана на использовании аппаратных ключей (токенов) и синхронизации по времени.
Аутентификация основана на генерации случайных чисел **через определенные временные интервалы (60 сек)**.
Уникальный **секретный ключ** хранится только в базе системы и в аппаратном устройстве субъекта.
Когда субъект запрашивает доступ в систему, ему предлагается ввести PIN-код (4 цифры), а также случайно генерируемое число, отображаемое **в этот момент** на аппаратном устройстве (токен-код 6 цифр).
Система сопоставляет введенный PIN-код и секретный ключ субъекта из своей базы и генерирует случайное число, основываясь на **параметрах секретного ключа из базы и текущего времени**.
Далее проверяется идентичность сгенерированного числа и числа, введенного субъектом (токен-кода)

Стоит переработать временные метки, тяжело читать

Использование единой базы паролей для субъекта и системы высокоточной синхронизации между ними.

Каждый пароль из наборе может быть использован только один раз.

28. Одноразовые пароли. Система S/Key генерации одноразовых паролей.

One-time Password (OTP)

Система S/Key для генерации одноразовых паролей

- Определена в RFC 1760.

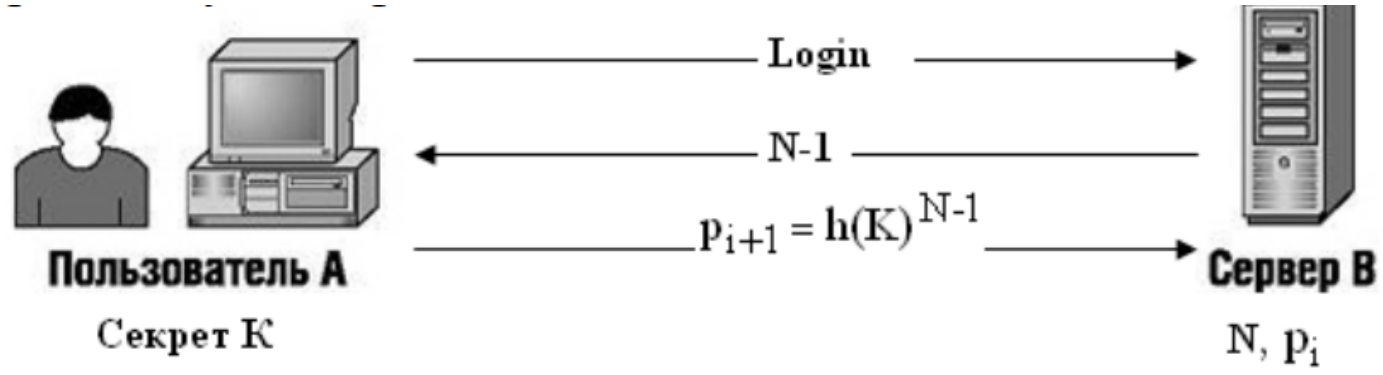
- представляет собой систему генерирования одноразовых паролей на основе стандартов `MD4` и `MD5`.
- предназначена для борьбы с "повторными атаками"

Архитектура

Генерация одноразовых паролей:

$$p_0 = h(K)^N, \quad p_1 = h(K)^{N-1}, \ldots, \quad p_i = h(K)^{N-i}, \ldots, p_{N-1} = h(K)$$

```
K - вектор инициализации
N - целое число
h - хэш-функция
0 <= i <= N-1
```



1. Пользователь посылает серверу свой логин
2. Сервер в ответ посылает число N-1
3. Пользователь вычисляет одноразовый пароль $p_{i+1} = h(K)^{N-1}$
4. Сервер вычисляет $p'_i = h(p_{i+1})$. Если $p'_i = p_i$ пользователь - подлинный.

Предполагается, что пользователю известна секретная величина k.
На предварительном этапе k объединяется со случайной величиной S, присланной сервером.
В результате получается некоторая величина K=(k||S)
Далее пользователь выбирает некоторое целое число N, лежащее в диапазоне от 500 до 1000.
Генерирует первый одноразовый пароль путем применения хэш-функции к значению (k||s) N раз.
Следующий одноразовый пароль будет генерироваться путем применения хэш-функции к значению (k||s) только (N - 1) раз и т.д.
Таким образом формируется последовательность одноразовых паролей, используемых для аутентификации
Пользователь посылает на сервер первое значение одноразового пароля P_0, вычисленное по

формуле и число N.
Данный пароль для аутентификации не используется, а выступает для сервера в роли начального значения, на основе которого будет проверяться следующий присланный пользователем во время аутентификации одноразовый пароль.

Достоинства

1. Защита от повторного использования
2. Вектор инициализации k не хранится на сервере и не передается по сети.

Недостатки

1. Ограниченное количество генерируемых паролей, заданных величиной N
2. При больших значениях N увеличивается время генерации одноразового пароля на стороне пользователя.
3. Отсутствует аутентификация сервера

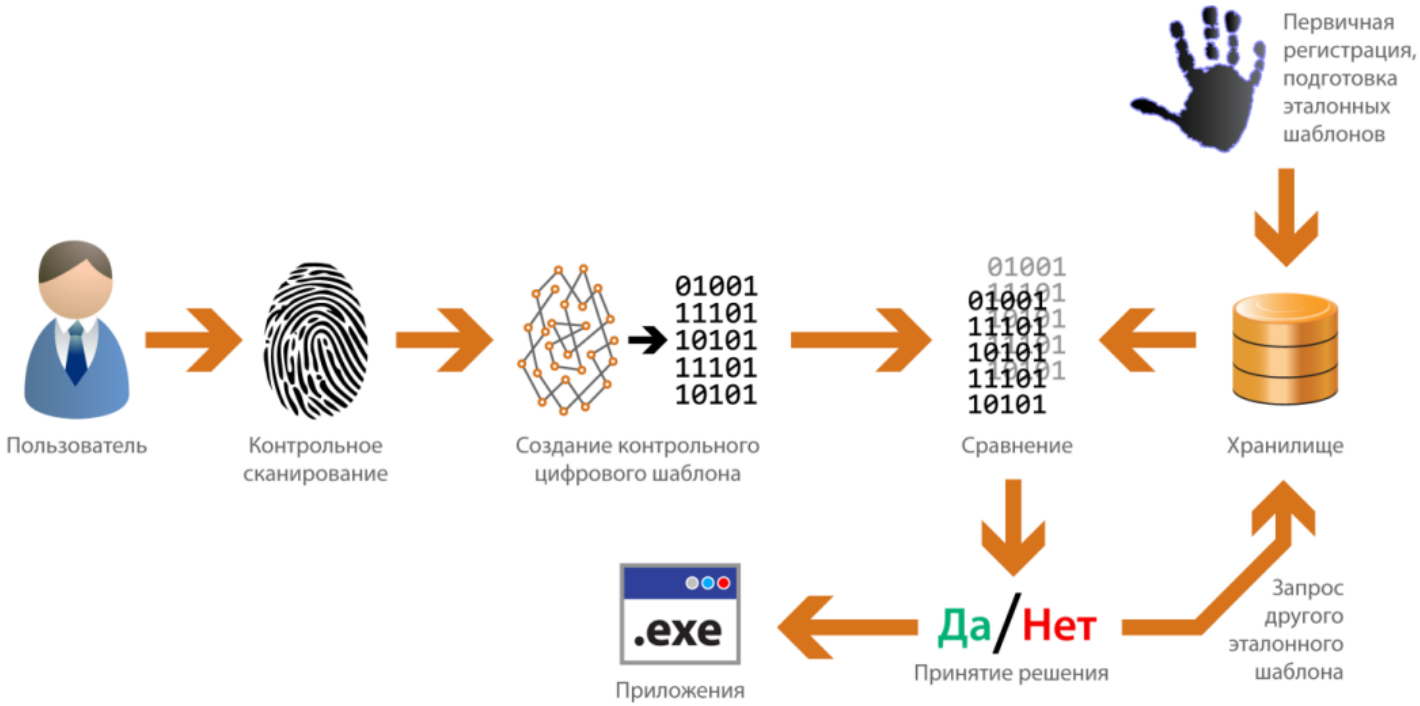
29. Охарактеризовать виды биометрической аутентификации.

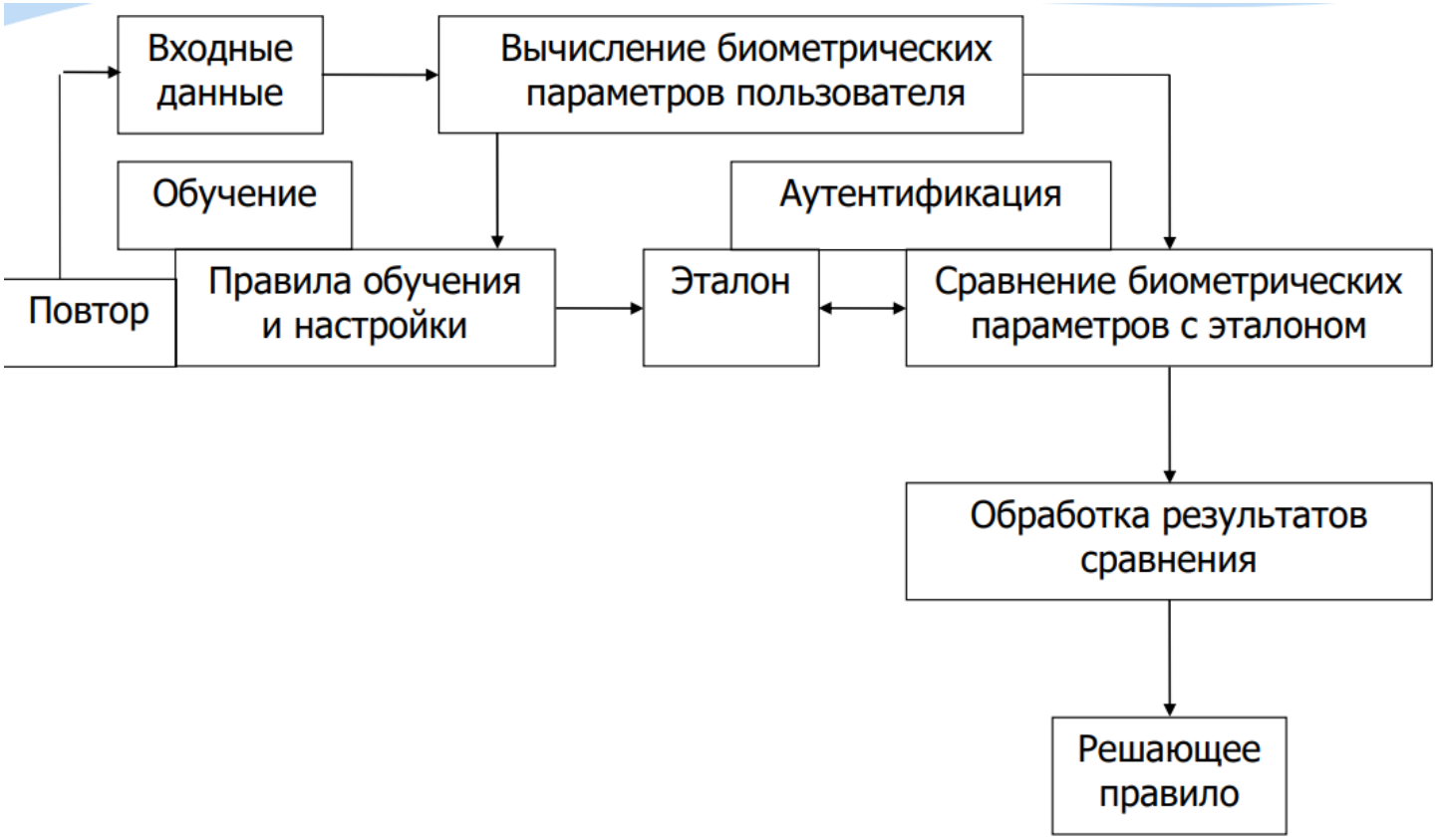
Виды:

1. **По отпечаткам пальцев**
2. **По геометрии кисти руки**
3. По отпечатку ладони
4. По строению кровеносных сосудов
5. По термографии лица
6. **По форме лица в 2-х, 3-х мерном измерении**
7. **По особенностям голоса**
8. По запаху
9. По подписи
10. По динамике печатания
11. По походке
12. **По радужной оболочке глаза**
13. По сетчатке глаза

30. Охарактеризовать структуру биометрических систем доступа

Основные элементы биометрической системы доступа





31. Аутентификационные факторы и их комбинации. Многофакторная аутентификация.

1. Электронный ключ + PIN-код
2. Электронный ключ + отпечаток пальца
3. Электронный ключ + PIN-код + отпечаток пальца

32. Охарактеризовать обеспечение подлинности сеанса связи с использованием механизма запроса-ответа.

Основные механизмы, используемые для подтверждения подлинности:

- запроса-ответа (используется для аутентификации участников)
- временной штемпель (используется для аутентификации связи)

Механизм запрос-ответ

1. Пользователь А (проверяющий) включает в посылаемое для В сообщение непредсказуемый элемент- запрос X (например, некоторое случайное число).
2. При ответе В должен вычислить некоторую функцию f(X).
3. Получив правильный ответ, А может быть уверен в подлинности В.

Недостаток – возможность установления закономерности между запросом и ответом, т.е. определения вида функции f.
Устранение – использование шифрования.

33. Охарактеризовать обеспечение подлинности сеанса связи с использованием механизма отметок времени (временного штемпеля)

Основные механизмы, используемые для подтверждения подлинности:

- запроса-ответа (используется для аутентификации участников)
- временной штемпель (используется для аутентификации связи)

Механизм отметки времени («временной штемпель»)

Механизм подразумевает *регистрацию времени для каждого сообщения*.
В этом случае **каждый** пользователь **может определить**, насколько **«устарело»** пришедшее сообщение, и решить не принимать его, так как это может быть повтор сообщения, потерявшего свою актуальность.
При использовании отметок времени возникает **проблема допустимого временного интервала задержки** для подтверждения подлинности.

34. Охарактеризовать схему взаимной аутентификации с использованием рукопожатия.

По сути это механизм запрос-ответ, усиленный шифрованием

Стороны признают друг друга законными партнерами, если докажут друг другу, что обладают правильными ключами.
Процедуру рукопожатия обычно применяют в компьютерных сетях при организации сеанса связи между:

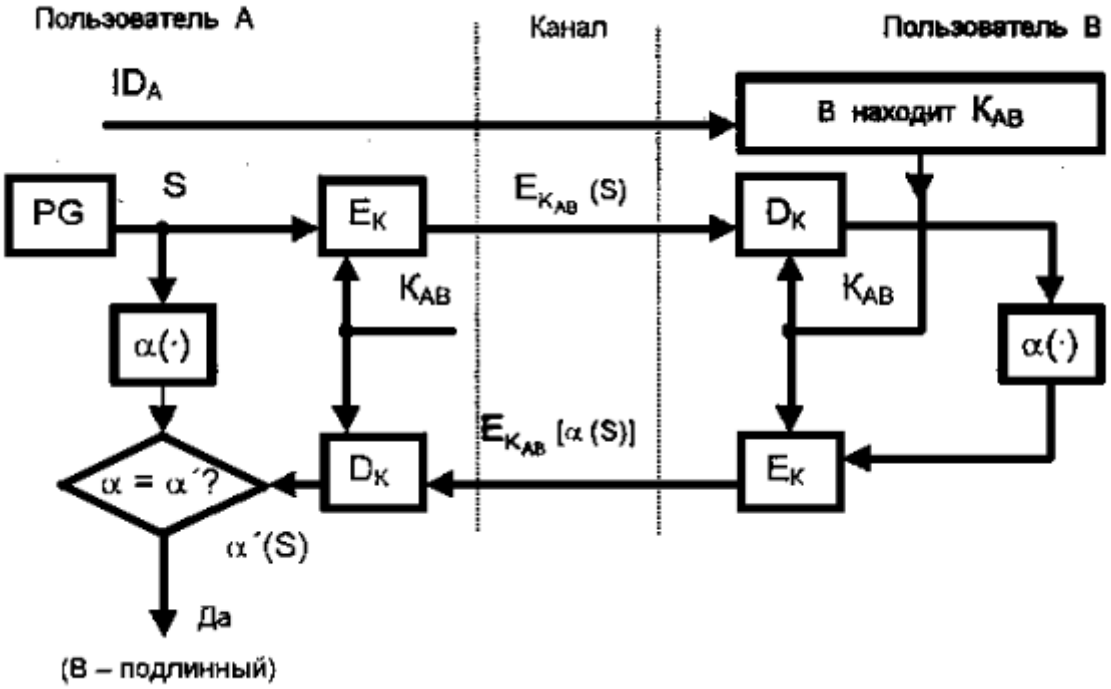
1. пользователями,
2. пользователем и хост-компьютером,
3. между хост-компьютерами и т.д.

Рассмотрим в качестве примера процедуру рукопожатия для двух пользователей A и B. (*Это допущение не влияет на общность рассмотрения*). Пусть применяется симметричная криптосистема. Пользователи A и B разделяют один и тот же секретный кпюч $К_{AB}$.

1. Пусть пользователь A инициирует процедуру рукопожатия, отправляя пользователю B свой идентификатор ID_A в открытой форме.
2. Пользователь B, получив идентификатор ID_A, находит в базе данных секретный ключ $К_{AB}$ и вводит его в свою криптосистему.
3. Тем временем пользователь A генерирует случайную последовательность S с помощью псевдослучайного генератора PG и отправляет ее пользователю B в виде **криптограммы**.
4. Пользователь B расшифровывает эту криптограмму и раскрывает исходный вид последовательности S.
5. Затем оба пользователя A и B преобразуют последовательность S, используя открытую одностороннюю функцию a(·).
6. Пользователь B шифрует сообщение a(S) и отправляет эту криптограмму пользователю A.
7. Наконец, пользователь A расшифровывает эту криптограмму и сравнивает полученное сообщение a'(S) с исходным a(S). Если эти сообщения равны, пользователь A признает подлинность пользователя B.

Очевидно, пользователь B проверяет подлинность пользователя A таким же способом. Обе эти процедуры образуют процедуру рукопожатия, которая обычно выполняется в самом начале любого сеанса связи между любыми двумя сторонами в компьютерных сетях.

Достоинством модели рукопожатия является то, что ни один из участников сеанса связи не получает никакой секретной информации во время процедуры подтверждения подлинности. Вся процедура показана на рис 1



35. Схема взаимной аутентификации с использованием рукопожатия. Аутентификация сторон взаимодействия с использованием механизма «запрос-ответ»

Похоже, что повтор вопроса 34.

По сути это механизм запрос-ответ, усиленный шифрованием

Стороны признают друг друга законными партнерами, если докажут друг другу, что обладают правильными ключами.
Процедуру рукопожатия обычно применяют в компьютерных сетях при организации сеанса связи между:

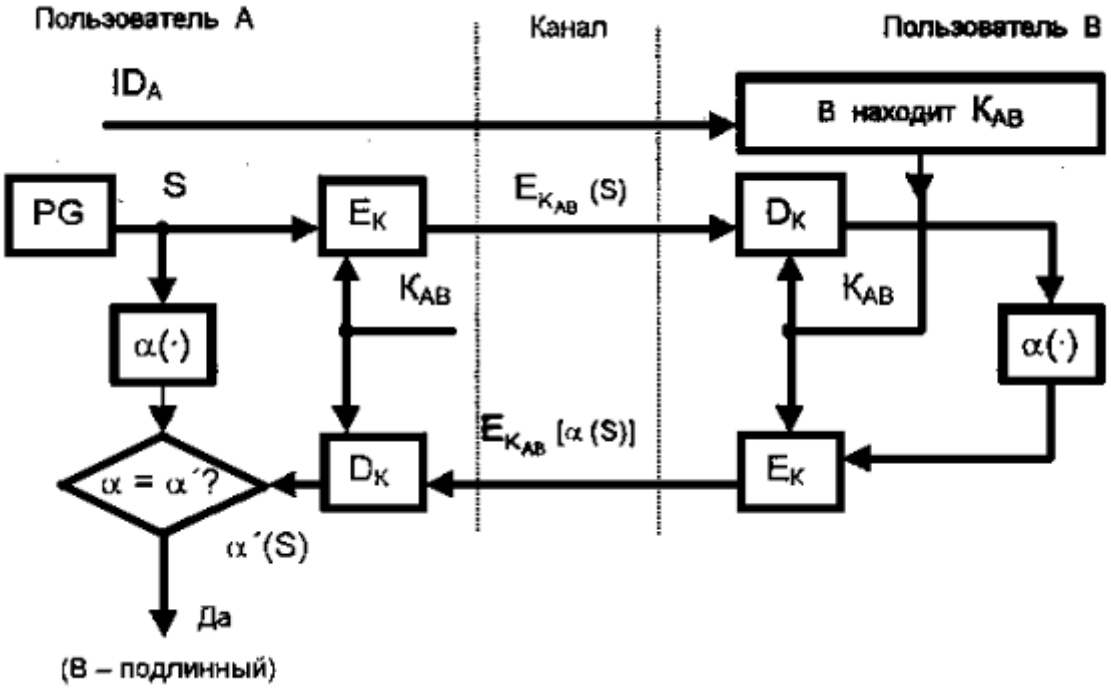
1. пользователями,
2. пользователем и хост-компьютером,
3. между хост-компьютерами и т.д.

Рассмотрим в качестве примера процедуру рукопожатия для двух пользователей А и В. (*Это допущение не влияет на общность рассмотрения*). Пусть применяется симметричная криптосистема. Пользователи А и В разделяют один и тот же секретный кпюч K_{AB}.

1. Пусть пользователь A инициирует процедуру рукопожатия, отправляя пользователю B свой идентификатор ID_A в открытой форме.
2. Пользователь B, получив идентификатор ID_A, находит в базе данных секретный ключ K_{AB} и вводит его в свою криптосистему.
3. Тем временем пользователь A генерирует случайную последовательность S с помощью псевдослучайного генератора PG и отправляет ее пользователю B в виде **криптограммы**.
4. Пользователь B расшифровывает эту криптограмму и раскрывает исходный вид последовательности S.
5. Затем оба пользователя A и B преобразуют последовательность S, используя открытую одностороннюю функцию a(·).
6. Пользователь B шифрует сообщение a(S) и отправляет эту криптограмму пользователю A.
7. Наконец, пользователь A расшифровывает эту криптограмму и сравнивает полученное сообщение a'(S) с исходным a(S). Если эти сообщения равны, пользователь A признает подлинность пользователя B.

Очевидно, пользователь B проверяет подлинность пользователя A таким же способом. Обе эти процедуры образуют процедуру рукопожатия, которая обычно выполняется в самом начале любого сеанса связи между любыми двумя сторонами в компьютерных сетях.

Достоинством модели рукопожатия является то, что ни один из участников сеанса связи не получает никакой секретной информации во время процедуры подтверждения подлинности. Вся процедура показана на рис 1



36. Охарактеризовать ошибки первого и второго рода при биометрической аутентификации.

Точность биометрической системы измеряется двумя параметрами

Ошибки первого рода | *Чувствительность*

```
False Rejection Rate
```

Коэффициент ложного отказа в доступе или вероятность того, что человек может быть не распознан системой.

Системы с ***низким значением FRR более комфортны*** для пользователей

Типичные значения FRR — порядка одной ошибки на 100.

Ошибки второго рода | *Специфичность*

```
False Acceptance Rate
```

Коэффициент ложного доступа или порог, определяющий вероятность того, что один человек может быть принят за другого.

Системы с ***низким значением FAR более защищены***.

Типичные значения FAR — порядка одной ошибки на 10 000.

37. Схема централизованного распределения ключей для симметричной криптосистемы. Аутентификация сторон взаимодействия.

Ключи хранятся в центре распределение ключей (ЦРК).
K_A - общий секретный ключ ЦРК и участника А
K_B - общий секретный ключ ЦРК и участника В
T – временная метка
L – срок действия сеансового ключа K_S
E – шифрование симметричное (например, по DES)
Ks – сеансовый ключ

1. A -> ЦРК : Id_A , Id_B
2. ЦРК -> A:
   - E_{Ka} (T, L, K_S, Id_B) ,
   - E_{Kb} (T, L, K_S, Id_A)
     - Сторона расшифровывает первую часть сообщения своим секретным ключом, проверяет метку времени T.
3. A -> B:
   - E_{Kb}(T, L, K_S, Id_A),
   - E_{Ks}(T', Id_A)
     - Сторона B расшифровывает обе части сообщения. Совпадение Id_A и T гарантирует достоверность пользователя A. Расшифровывает первую часть сообщения своим секретным ключом и берет функцию от T.
4. B -> A:
   - E_{Ks}(f(T)) – по схеме запрос ответ
     - Сторона B отсылает стороне A функцию от T, зашифрованную на сеансовом ключе. Сторона A расшифровывает сообщение и проверяет совпадают ли f(T) и f(T')

Из презентаций

СХЕМА ЦЕНТРАЛИЗОВАННОГО РАСПРЕДЕЛЕНИЯ ОТКРЫТЫХ КЛЮЧЕЙ (1)

A – инициатор, запрашивает выдачу сертификатов A и B.

1. A → ЦРК: Id_A, Id_B «Пришлите сертификаты A и B»

2. ЦРК → A: ЦРК передает A два сертификата:

$C_A = E_K{}^C{}_{ЦРК}\,(h\,(L_A\,, K^O{}_A, Id_A)), (L_A\,, K^O{}_A, Id_A); C_A = C_A\ \{L_A\,, K^O{}_A, Id_A\ \}$

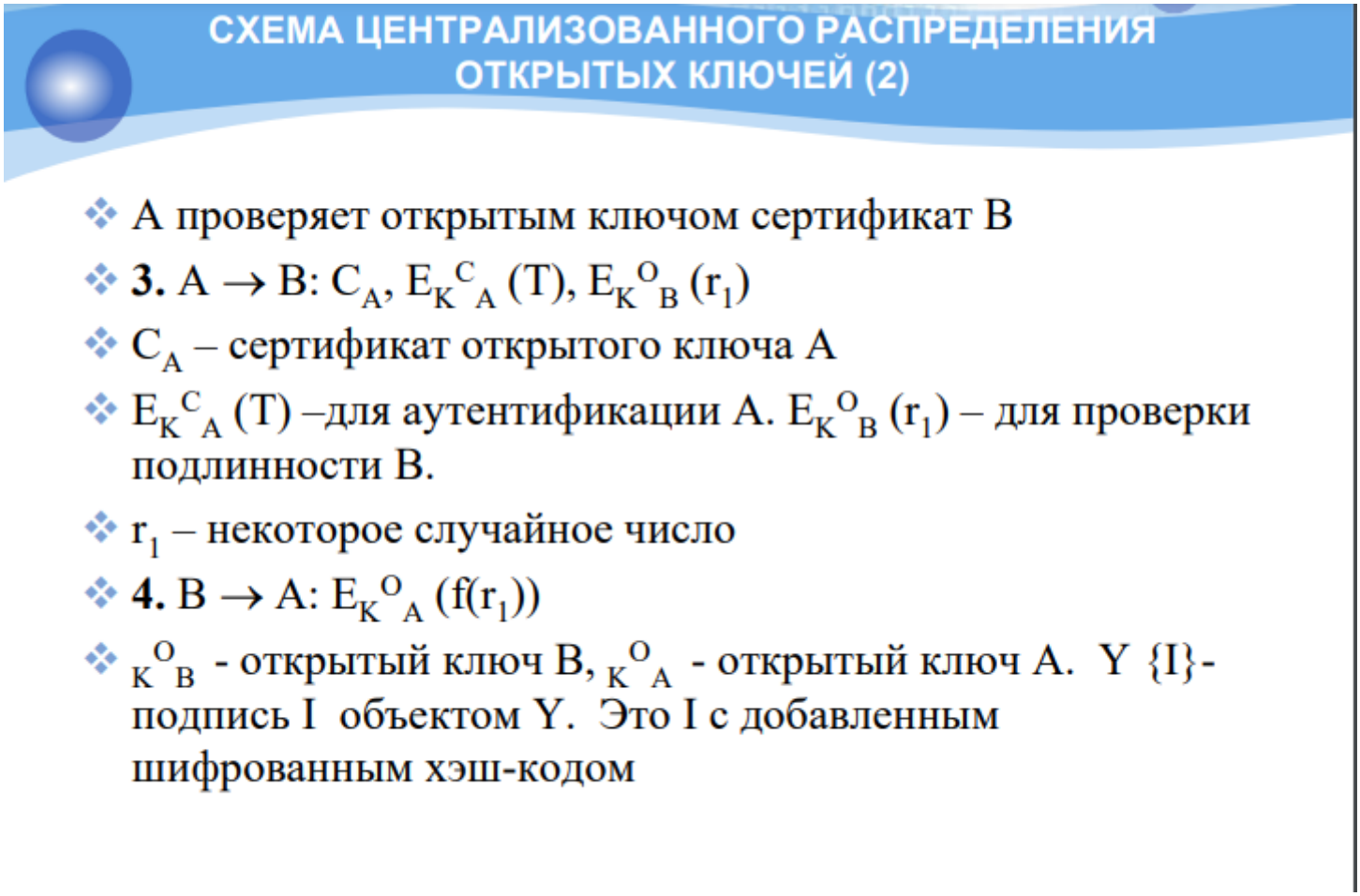
$C_B = E_K{}^C{}_{ЦРК}\,(h(L_B\,, K^O{}_B, Id_B)), (L_B\,, K^O{}_B, Id_B).\ C_B = C_B\ \{L_B\,, K^O{}_B, Id_B\ \}$

A проверяет подлинность сертификата B и берет себе $K^O{}_B$. Свой ОК у него есть. Проверяет сертификат B путем:

- Проверить подпись;
- Проверить сроки L_A, L_B действия сертификатов C_A,C_B.

Успешная проверка подписи говорит о том, что информация подписана ЦРК и что ключ B $K^0{}_B$ – подлинный.

Проверка сроков L_A, L_B используется для подтверждения актуальности сертификатов.



38. Базовый протокол распределения ключей для асимметричных криптосистем с использованием сертификатов открытых ключей.

В этом протоколе используется идея сертификатов открытых ключей.

Сертификатом открытого ключа С называется сообщение ЦРК, удостоверяющее целостность некоторого открытого ключа объекта. Например, сертификат открытого ключа для пользователя А, обозначаемый СА, содержит отметку времени Т, идентификатор IdA и открытый ключ КА, зашифрованные секретным ключом ЦРК КЦРК, т. е. СА = Екцрк(Т, IdA, КА).

Отметка времени Т используется для подтверждения актуальности сертификата и тем самым предотвращает повторы прежних сертификатов, которые содержат открытые ключи и для которых соответствующие секретные ключи несостоятельны.

Секретный ключ кЦРК известен только менеджеру ЦРК. Открытый ключ Кцрк известен участникам А и В. ЦРК поддерживает таблицу открытых ключей всех объектов сети, которые он

обслуживает.

Вызывающий объект А инициирует стадию установления ключа, запрашивая у ЦРК сертификат своего открытого ключа и открытого ключа участника В:

1. **А -> ЦРК: IdA,IdB ,"Вышлите сертификаты ключей А и В".**

Здесь IdA и IdB- уникальные идентификаторы соответственно участников А и В.

Менеджер ЦРК отвечает сообщением

2. **ЦРК-> А: Екцрк(Т, IdA. Ка), Екцрк(Т, IdB, КВ).**

Участник А, используя открытый ключ ЦРК кЦРК, расшифровывает ответ ЦРК, проверяет оба сертификата. Идентификатор IdB убеждает А, что личность вызываемого участника правильно зафиксирована в ЦРК и Кв - действительно открытый ключ участника В, поскольку оба зашифрованы ключом кЦРК.

Хотя открытые ключи предполагаются известными всем, посредничество ЦРК позволяет подтвердить их целостность. Без такого посредничества злоумышленник может снабдить А своим открытым ключом, который А будет считать ключом участника В. Затем злоумышленник может подменить собой В и установить связь с А, и его никто не сможет выявить. Следующий шаг протокола включает установление связи А с В:

3. **А->В: СА1 Ека(Т),Екв(г1).**

Здесь СА-сертификат открытого ключа пользователя А; Ека(Т)-отметка времени, зашифрованная секретным ключом участника А и являющаяся подписью участника А, поскольку никто другой не может создать такую подпись; г - случайное число, генерируемое А и используемое для обмена с В в ходе процедуры подлинности.

Если сертификат СА и подпись А верны, то участник В уверен, что сообщение пришло от А. Часть сообщения Екв(г,) может расшифровать только В, поскольку никто другой не знает секретного ключа кв, соответствующего открытому ключу Кв. Участник В расшифровывает значение числа г, и, чтобы подтвердить свою подлинность, посылает участнику А сообщение

4. **В->Ека(r,).**

Участник А восстанавливает значение г, расшифровывая это сообщение с использованием своего секретного ключа кА. Если это ожидаемое значение г, то А получает подтверждение, что вызываемый участник действительно В. Протокол, основанный на симметричном шифровании, функционирует быстрее, чем протокол, основанный на криптосистемах с открытыми ключами.

Однако способность систем с открытыми ключами генерировать цифровые подписи, обеспечивающие различные функции защиты, компенсирует избыточность требуемых вычислений.

39. Инфраструктура открытых ключей в соответствие с рекомендациями X.509.

Инфраструктура открытых ключей (PKI - Public Key Infrastructure) - технология аутентификации с помощью открытых ключей. Это комплексная система, которая связывает открытые ключи с личностью пользователя посредством удостоверяющего центра (УЦ).

Фактически, PKI представляет собой систему, основным компонентом которой является удостоверяющий центр и пользователи, взаимодействующие между собой посредством удостоверяющего центра.

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

- закрытый ключ известен только его владельцу;
- удостоверяющий центр создает сертификат открытого ключа, 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.
Основные компоненты PKI:

- **Удостоверяющий центр (УЦ)** является основной структурой, формирующей цифровые сертификаты подчиненных центров сертификации и конечных пользователей.
- **Сертификат открытого ключа (чаще всего просто сертификат)** - это данные пользователя и его открытый ключ, скрепленные подписью УЦ.
- **Certificate Authority (CA) – удостоверяющий (сертификационный) центр** – это уполномоченный орган, который создает и подписывает сертификаты открытого ключа. Дополнительно CA может создавать пары закрытый/открытый ключ конечного участника. Важно заметить, что CA отвечает за сертификаты открытого ключа в течение всего времени их жизни, а не только в момент выпуска.
- **Регистрационный центр (РЦ)** - необязательный компонент системы, предназначенный для регистрации пользователей. Удостоверяющий центр доверяет регистрационному центру

проверку информации о субъекте. Регистрационный центр проверив правильность информации, подписывает её своим ключом и передаёт удостоверяющему центру, который, проверив ключ регистрационного центра, выписывает сертификат.

- Один регистрационный центр может работать с несколькими удостоверяющими центрами (т.е. состоять в нескольких PKI), один удостоверяющий центр может работать с несколькими регистрационными центрами.
  Иногда, удостоверяющий центр выполняет функции регистрационного центра.
- **Конечные пользователи** - пользователи или приложения, являющиеся владельцами сертификата и использующие инфраструктуру управления открытыми ключами.

X.500 – стандарт службы каталогов.
Рекомендации **X.509** Международного союза телекоммуникаций (ITU – International Telecommunication Union) – часть рекомендаций серии X.500. Появился в 1988 году. После исправлений – в 1993 году.

Обозначения:

- Y«X» - удостоверение пользователя X, выданное центром сертификации Y
- Y{I} – подпись I объектом Y. Она состоит из I с добавленным шифрованным хэш-кодом.

Основные поля сертификата X.509:

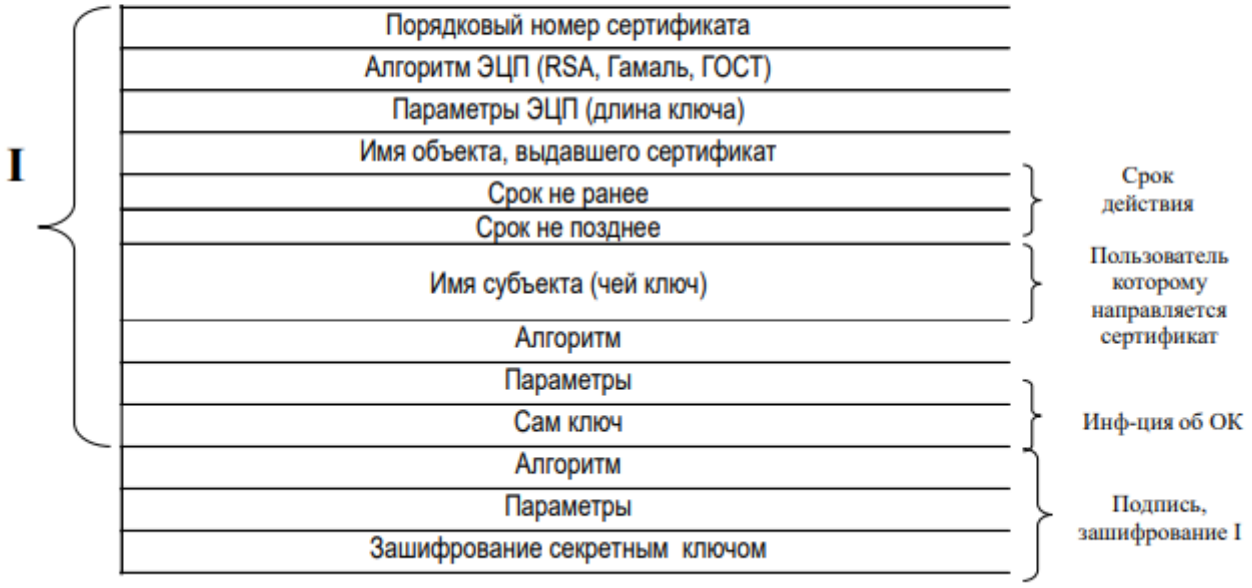
- V – версия:
- SN – порядковый номер;
- AI – идентификатор алгоритма подписи (не слишком полезное, т.к. в конце есть поле в подписи);
- CA – имя объекта, выдавшего сертификат;
- T_A – срок действия;
- A – имя субъекта;
- A_P – информация об открытом ключе субъекта.

CA«A» = CA {V,SN,AI,CA,T_A,A,A_P}, где
Y«X» - удостоверение X, выдан. Y; Y{I} – подпись I объектом Y

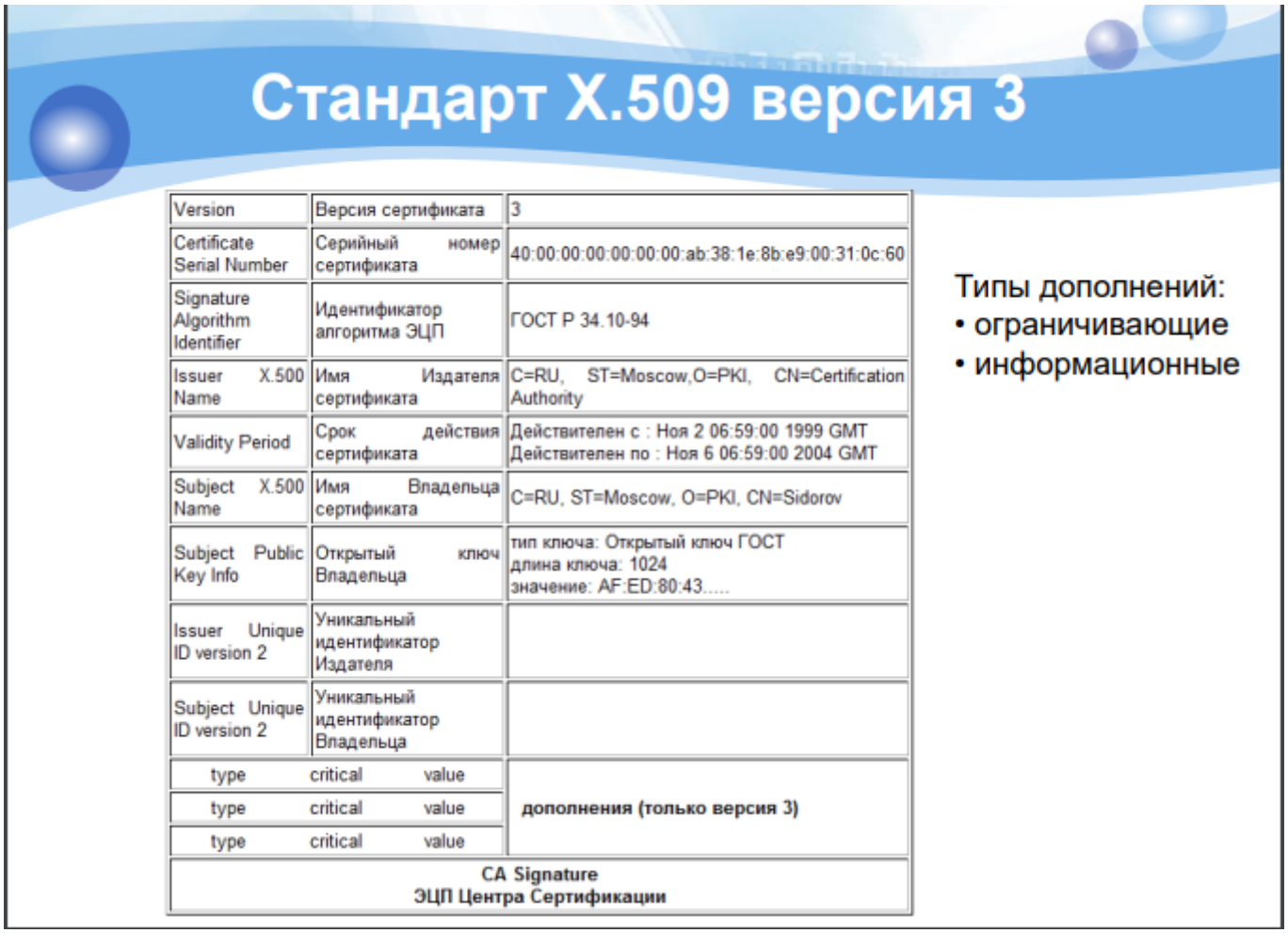
Структура сертификата X.509

Структура сертификата X.509

Рекомендация ITU – international telecommunicaon Union



Стандарт X.509 версия 3



Отзыв сертификатов

В некоторых ситуациях желательно иметь возможность отменить действие сертификата до окончания срока его действия по следующим причинам:

- Секретный ключ пользователя оказался скомпрометированным.
- Пользователь больше не сертифицируется в данном центре сертификации.
- Сертификат данного центра сертификации оказался скомпрометированным.

Каждый центр должен поддерживать **список отозванных сертификатов (CRL – Certificate Revocation List)**. **CRL** должны размещаться в каталоге, подписываются центром сертификации и включают имя центра, дату создания списка, дату выхода следующей версии CRL и **запись** для каждого отозванного сертификата. **Запись** состоит из
порядкового номера сертификата и даты отзыва этого сертфиката.

Важность стандарта X.509

Структура сертификатов и протоколов аутентификации, определяемых в X.509, используется в протоколах
S/MIME (Secure/Multipurpose Internet Mail Extension) – защищенное многоцелевое расширение электронной почты, IP Security, SSL/TLS, SET.

40. Понятие сертификата открытого ключа. Принципы работы удостоверяющих центров.

Сертификат открытого ключа (чаще всего просто сертификат) - это данные пользователя и его открытый ключ, скрепленные подписью УЦ.
Certificate Authority (CA) – удостоверяющий (сертификационный) центр – это уполномоченный орган, который создает и подписывает сертификаты открытого ключа. Дополнительно CA может создавать пары закрытый/открытый ключ конечного участника. Важно заметить, что CA отвечает за сертификаты открытого ключа в течение всего времени их жизни, а не только в момент выпуска.

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

- закрытый ключ известен только его владельцу;
- удостоверяющий центр создает сертификат открытого ключа, 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.

Удостоверяющий центр устанавливает определенные требования к работе пользователей. Например, удостоверяющий центр определяет максимальный срок действия сертификатов, совокупность необходимых данных запросе на сертификат, способы передачи запроса от пользователя в УЦ, способы проверки корректности запросов пользователей и т. д. Совокупность требований удостоверяющего центра называется **регламентом удостоверяющего центра**.

- Удостоверяющий центр имеет собственные ключи подписи и подписывает на них все электронные документы, которые он выпускает.
- Удостоверяющий центр выпускает сертификат на собственный открытый ключ. Такой сертификат называется сертификатом удостоверяющего центра.
  Таким образом, каждый пользователь в любой момент может, воспользовавшись сертификатом удостоверяющего центра, проверить корректность любого сертификата.

Взаимодействие пользователя с удостоверяющим центром происходит следующим образом:

- Пользователь создает ключевую пару (открытый и закрытый ключи).
- Пользователь отправляет в удостоверяющий центр запрос на сертификат, в который включает открытый ключ и всю необходимую информацию о себе и о ключах. Набор

необходимых сведений определяется регламентом удостоверяющего центра, но всегда необходимо указывать имя владельца, назначение ключей, дату создания.
- Удостоверяющий центр получает запрос и проверяет его подлинность и корректность. Как именно это делается, определяется регламентом удостоверяющего центра.
- Если результат проверки запроса положительный, удостоверяющий центр создает сертификат на открытый ключ, подписывает его, заносит в свою базу данных и отправляет пользователю.
- Пользователь получает сертификат и устанавливает его у себя в системе.

Удостоверяющий центр и пользователи, чьи сертификаты зарегистрированы в удостоверяющем центре, вместе составляют криптосеть.

41. Структура сертификата по рекомендациям X.509.

Основные поля сертификата X.509:

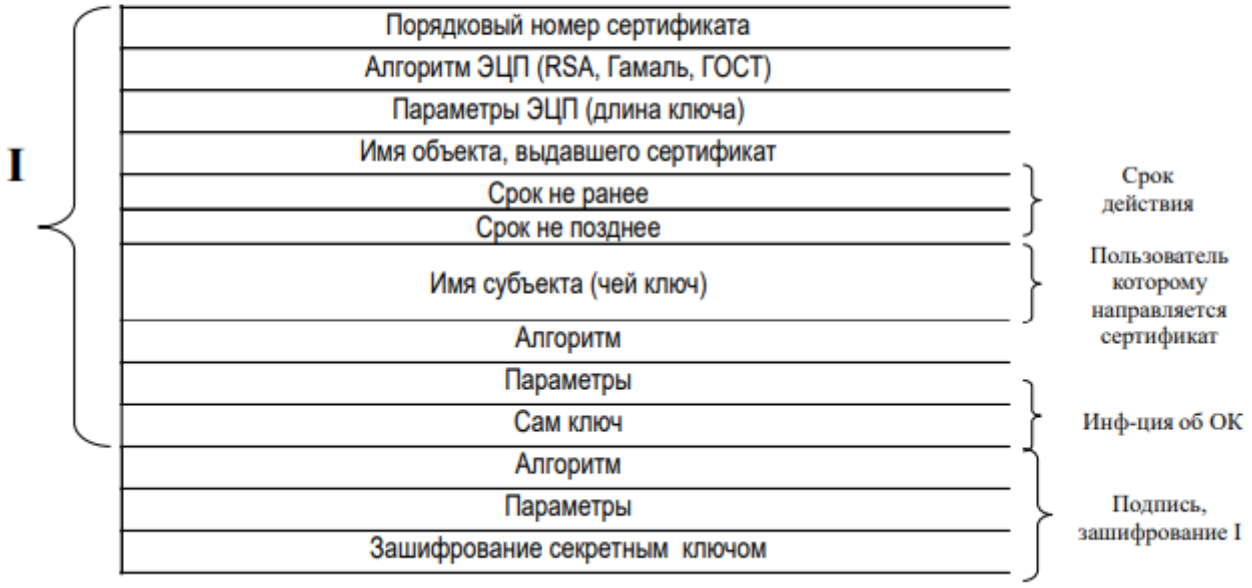
- V – версия:
- SN – порядковый номер;
- AI – идентификатор алгоритма подписи (не слишком полезное, т.к. в конце есть поле в подписи);
- CA – имя объекта, выдавшего сертификат;
- T_A – срок действия;
- A – имя субъекта;
- A_P – информация об открытом ключе субъекта.

CA«A» = CA {V,SN,AI,CA,T_A,A,A_P}, где
Y«X» - удостоверение X, выдан. Y; Y{I} – подпись I объектом Y

Структура сертификата X.509

Структура сертификата X.509

Рекомендация ITU – international telecommunicaon Union



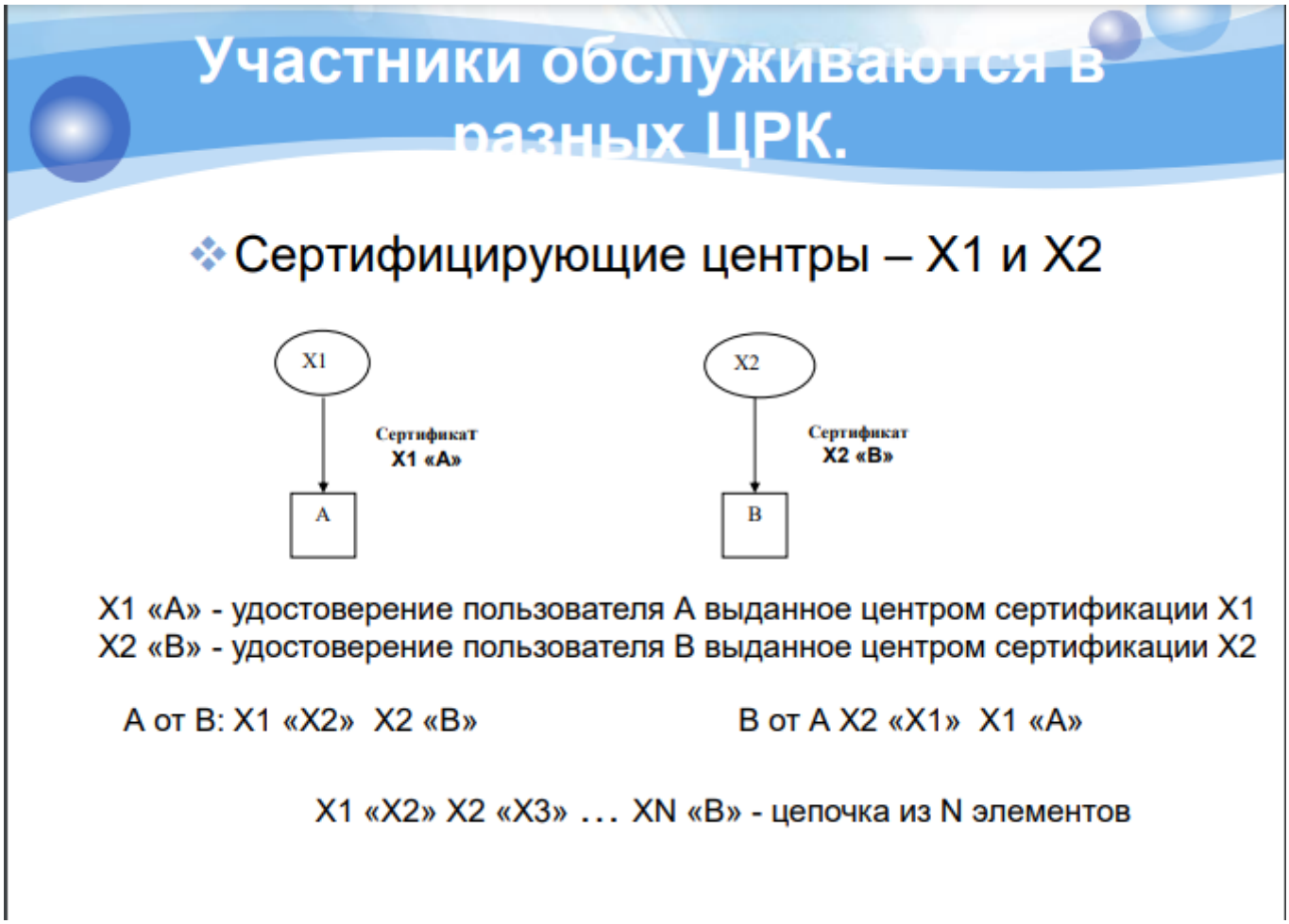
Стандарт X.509 версия 3

Стандарт X.509 версия 3

Version	Версия сертификата	3
Certificate Serial Number	Серийный номер сертификата	40:00:00:00:00:00:00:ab:38:1e:8b:e9:00:31:0c:60
Signature Algorithm Identifier	Идентификатор алгоритма ЭЦП	ГОСТ Р 34.10-94
Issuer X.500 Name	Имя Издателя сертификата	C=RU, ST=Moscow,O=PKI, CN=Certification Authority
Validity Period	Срок действия сертификата	Действителен с : Ноя 2 06:59:00 1999 GMT Действителен по : Ноя 6 06:59:00 2004 GMT
Subject X.500 Name	Имя Владельца сертификата	C=RU, ST=Moscow, O=PKI, CN=Sidorov
Subject Public Key Info	Открытый ключ Владельца	тип ключа: Открытый ключ ГОСТ длина ключа: 1024 значение: AF:ED:80:43.....
Issuer Unique ID version 2	Уникальный идентификатор Издателя	
Subject Unique ID version 2	Уникальный идентификатор Владельца	
type critical value		**дополнения (только версия 3)**
type critical value		
type critical value		
CA Signature ЭЦП Центра Сертификации		

Типы дополнений:
- ограничивающие
- информационные

42. Проверка сертификатов, в том числе полученных в разных удостоверяющих центрах.



X1 «A» - удостоверение пользователя A выданное центром сертификации X1
X2 «B» - удостоверение пользователя B выданное центром сертификации X2

A от B: X1 «X2» X2 «B»
X2 передается доверительным способом в X1 и подписывается стороной X1 (это как бы подпись). После того как X2 подписан, передается сам сертификат X2 «B».

B от A X2 «X1» X1 «A»
X1 «X2» X2 «X3» … XN «B» - цепочка из N элементов

Прямые сертификаты. Сертификаты X, выданные другими центрами сертификации.

Возвратные сертификаты. Сертификаты, выданные X для сертификации других центров сертификации.

Самоподписанный серитфикат. Открытый ключ для корневой подписи распространяется с автоподписью. Известен всем программным средствам.

43. Примеры современных программно-аппаратных средств реализации инфраструктуры открытых ключей (системы удостоверяющих и регистрационных центров).

ПО ViPNet KC & CA (Удостоверяющий и ключевой центр, УКЦ) реализует полный набор функций по управлению сертификатами открытых ключей:

- формирование пар открытый-закрытый ключ по запросам пользователей;
- изготовление сертификатов открытых ключей;

- приостановление и возобновление действия сертификатов, отзыв (аннулирование) сертификатов;
- ведение реестра (справочника) выпущенных сертификатов и списка отозванных сертификатов;
- поддерживает все архитектуры PKI.
-

ПАК «КриптоПро УЦ» обеспечивает организационно-техническую реализацию Удостоверяющего центра, предоставляя пользователям все необходимые средства и спецификации для использования сертификатов открытых ключей.
Обеспечивает:

- Реализацию инфраструктуры Удостоверяющих Центров, построенных как по иерархической так и по сетево(распределенной) модели.
- Генерацию ключей подписи и шифрования.
- Выполнение процедуры подтверждения подлинности ЭЦП.
- Ведения реестра зарегистрированных пользователей.
- Приостановление/возобновление действия сертификатов открытых ключей.

Атликс HSM в первую очередь предназначен для обеспечения безопасного хранения и использования закрытого ключа уполномоченного лица удостоверяющего центра, что обеспечивается выполнением всех криптографических операций, в том числе по генерации ключа уполномоченного лица в криптомодуле. Защита ключа уполномоченного лица удостоверяющего центра обеспечивается в том числе с использованием "раздельных секретов".

44. ViPNet PKI продукты.

ViPNet Registration Point (пункт регистрации) Пункт регистрации, выступая в качестве филиала УЦ, проводит идентификацию пользователей, генерирует для них ключевые пары или устанавливает факт владения закрытым ключом по предъявленному открытому ключу, после чего формирует и передает запрос на сертификацию в УЦ. Перенос части функций УЦ в пункт регистрации позволяет снизить требования по организационной, физической и информационной безопасности, что уменьшает расходы на создание УЦ. Пункты регистрации снижают нагрузку на УЦ по обработке запросов пользователей.
ПО ViPNet KC & CA (Удостоверяющий и ключевой центр, УКЦ) реализует полный набор функций по управлению сертификатами открытых ключей:

- формирование пар открытый-закрытый ключ по запросам пользователей;
- изготовление сертификатов открытых ключей;
- приостановление и возобновление действия сертификатов, отзыв (аннулирование) сертификатов;
- ведение реестра (справочника) выпущенных сертификатов и списка отозванных сертификатов;
- поддерживает все архитектуры PKI.

ViPNet CryptoService предоставляет пользователю возможность управлять своими криптографическими ключами: генерировать пары открытый-закрытый ключ, записывать ключи в защищенные контейнеры и внешние электронные носители и считывать ключи из них, обновлять сертификаты. Обмен ключевой и служебной информацией с компонентами PKI, созданными на базе ПО ViPNet, полностью автоматизирован и производится криптографически защищенным способом.

ViPNet Client (Клиент) — это программный комплекс для ОС Windows 2000/Windows XP/Vista/Windows 7/Server 2003/Server 2008 (32 бит), ОС Vista/Windows 7/Server 2008/Server 2008 R2 (64 бит), выполняющий на рабочем месте пользователя или сервере с прикладным ПО функции VPN-клиента, персонального экрана, клиента защищенной почтовой системы, а также криптопровайдера для прикладных программ, использующих функции подписи и шифрования.

45. Составляющие Удостоверяющего центра на базе «КриптоПро УЦ»

ПАК «КриптоПро УЦ» обеспечивает организационно-техническую реализацию Удостоверяющего центра, предоставляя пользователям все необходимые средства и спецификации для использования сертификатов открытых ключей.

Предназначен для:

- автоматизации деятельности Удостоверяющего Центра при выполнении им своих целевых функций согласно действующего законодательства РФ;
- автоматизации деятельности по управлению сертификатами открытых ключей, применяемых для шифрования, аутентификации и обеспечения достоверности информации.

Обеспечивает:

- Реализацию инфраструктуры Удостоверяющих Центров, построенных как по иерархической так и по сетевой(распределенной) модели.
- Генерацию ключей подписи и шифрования.
- Выполнение процедуры подтверждения подлинности ЭЦП.
- Ведения реестра зарегистрированных пользователей.
- Приостановление/возобновление действия сертификатов открытых ключей.

Основные составляющие Удостоверяющего центра на базе «КриптоПро УЦ»:

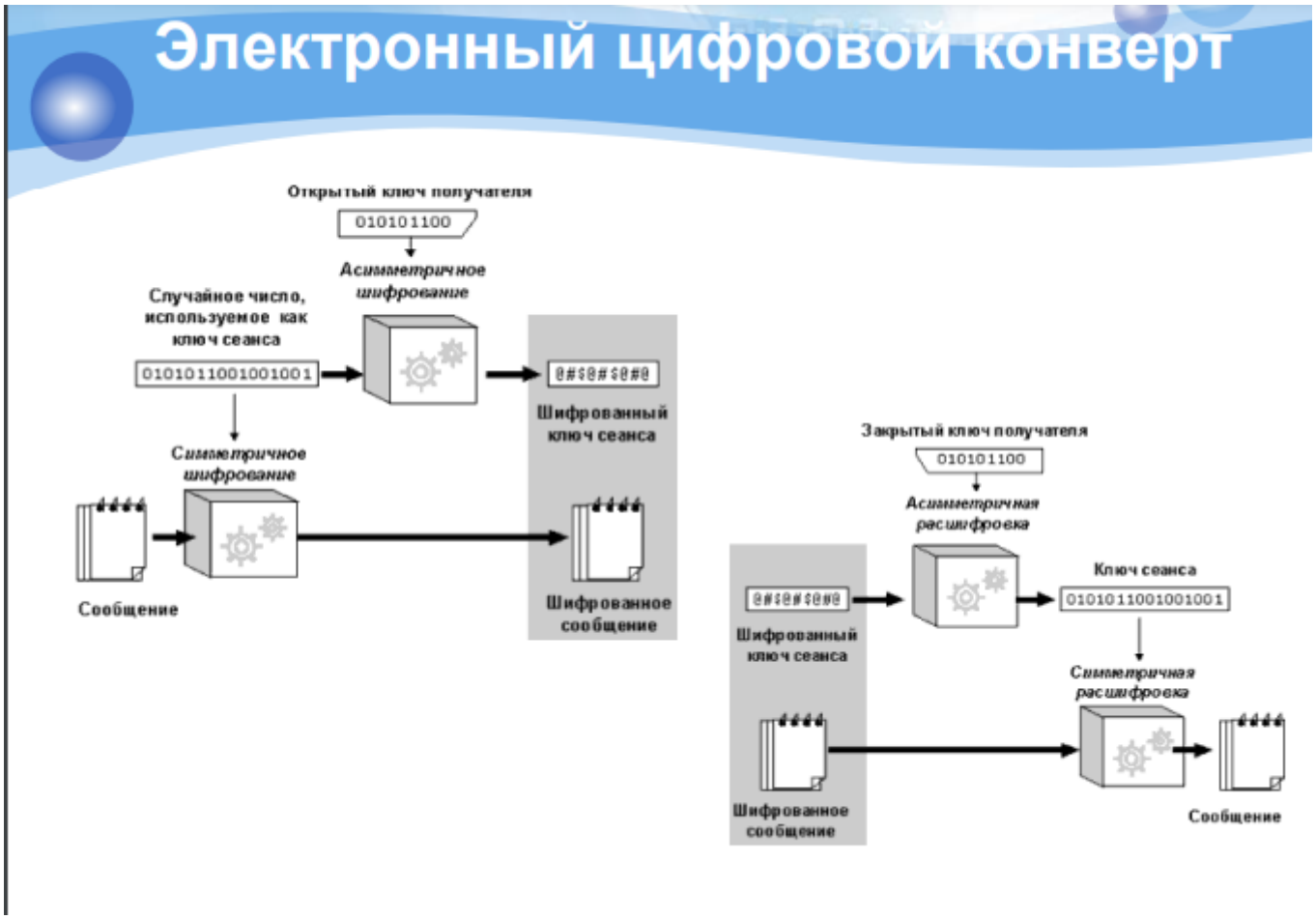
- Центр Сертификации «КриптоПро УЦ»;
- Центр Регистрации «КриптоПро УЦ»;
- АРМ пользователя в составе Центра Регистрации;
- АРМ администратора Центра Регистрации;
- АРМ разбора конфликтных ситуаций

Технические характеристики

- Создание и проверка электронной цифровой подписи (ЭЦП) - ГОСТ Р 34.10-2001, ГОСТ Р 34.11-94
- Шифрование и имитозащита - ГОСТ 28147-89 с использованием СКЗИ «КриптоПро CSP» версии 3.6, СКЗИ «КриптоПро CSP» версии 3.6.1. и ПАКМ «Атликс HSM»
- Формирование электронных сертификатов открытых ключей - x.509v3 (согласно RFC 3280 и RFC 5280 с учётом RFC 4491)

46. Прямой обмен ключами между пользователями с использованием криптосистемы с открытым ключом для

шифрования и передачи секретного ключа симметричной системы (электронный цифровой конверт).



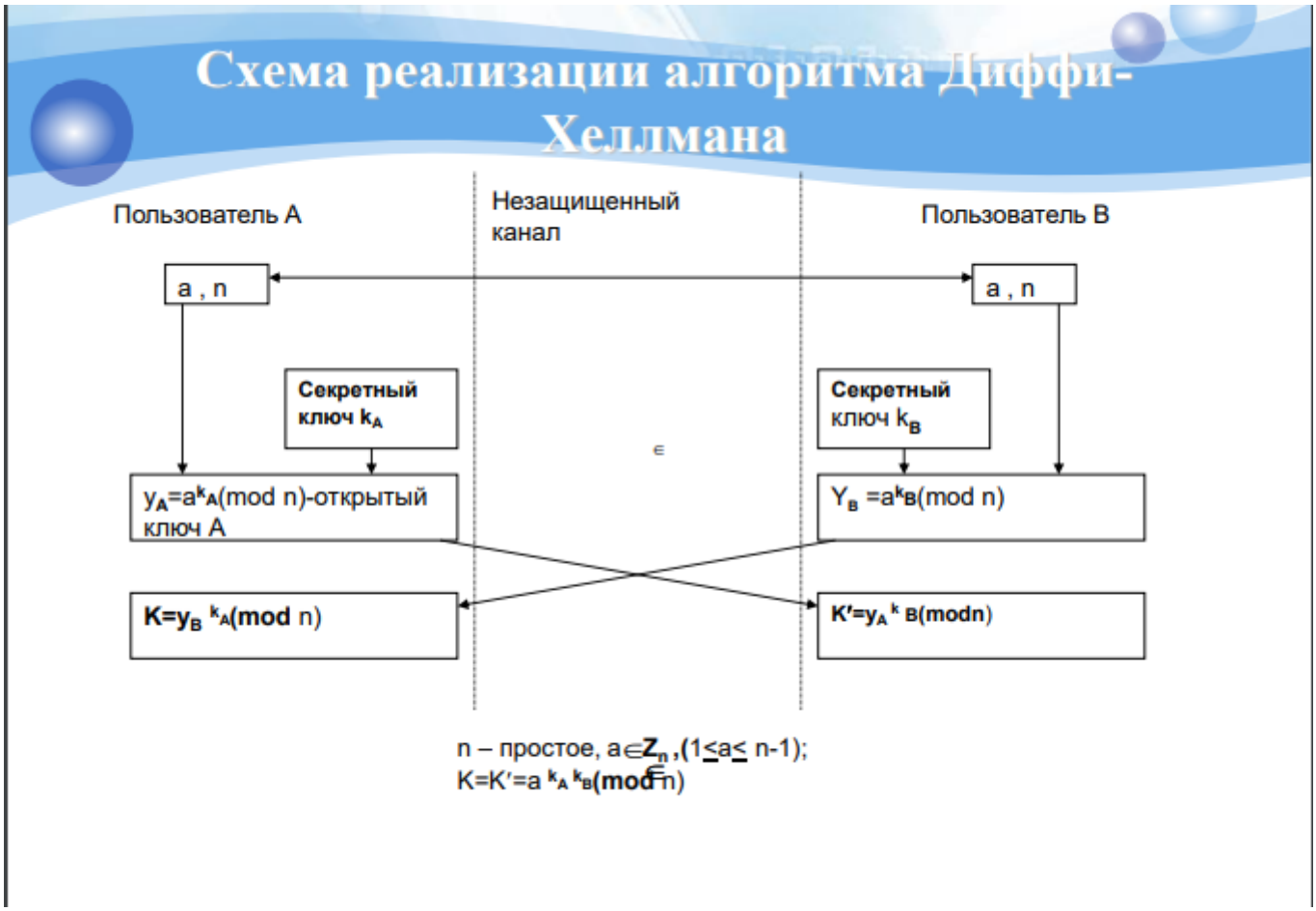
Электронный цифровой конверт

- A ЭЦК → B

- M – сообщение $K^c=D_{Kв}{}^c(C_к)$
- $C_m(M)=E_K{}^c(M)$ $M=D_k{}^c(C_m)$
- $C_k(K^c)=E_{Kв}{}^o(K^c)$

- E-шифрование, D-расшифрование,
- K^c-симметричный секретный ключ, $K_в{}^o$ – открытый ключ B, $K_в{}^c$–секретный ключ B.

47. Использование системы открытого распределения ключей Диффи-Хеллмана для формирования ключей.

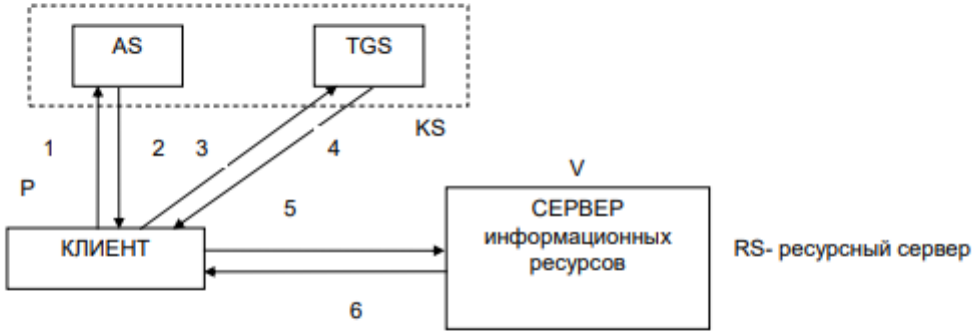


48. Симметричные методы аутентификации субъекта. Схема Kerberos.

СХЕМА АУТЕНТИФИКАЦИИ KERBEROS

СХЕМА АУТЕНТИФИКАЦИИ KERBEROS

(СИММЕТРИЧНАЯ)



AS – Аутентифицирующий сервер (типа ЦРК)
TGS – сервер выдачи разрешений (мандатов)
KS – сервер KERBEROS
K_{PG} – ключ для взаимодействия P и TGS (сеансовый)
K_{PV} – ключ для взаимодействия P и V

ПРОТОКОЛ СИСТЕМЫ KERBEROS

ПРОТОКОЛ СИСТЕМЫ KERBEROS

1. P→AS : P, V-запрос разрешить обратиться к G Примеч.: G обозначает TGS, tkt_{PG} → TGT
2. AS →P: E_P (T_{PG}, L_{PG}, K_{PG}, G, tkt_{PG})- разрешение обратиться к TGS

   $tkt_{PG} = E_G$ (T_{PG}, L_{PG}, K_{PG}, P)
3. P→ TGS : E_{PG} (T_{PG}, P), V, tkt_{PG}- запрос на допуск к RS

   E_{PG} (T_{PG}, P) – аутентифицирующая информация – удостоверение.
4. TGS →P: E_{PG} (T_{PV}, L_{PV}, K_{PV}, V, tkt_{PV})- разрешение на допуск к RS

   $tkt_{PV} = E_V$ (T_{PV}, L_{PV}, K_{PV}, P)

ПРОТОКОЛ СИСТЕМЫ KERBEROS

5.P $\rightarrow$V: E_{pv} (T_{pv},P), tkt_{pv} – запрос на получение информационного ресурса от RS

6.V$\rightarrow$P: E_{pv} (T_{pv}) или E_{pv} (f(T_{pv}))= E_{pv} (T_{pv}-1) подтв. подл.V и получение информационного ресурса

Обозначения

T_{pg} - отметка времени при направлении информации от P к TGS

T_{pv} - отметка времени при направлении информации от P к V

E_V -шифрование на ключе, который знает только V и центр (AS)

E_G -шифрование на ключе, который знает только TGS и центр (AS)

E_P -шифрование на ключе, который знает только P и центр (AS)

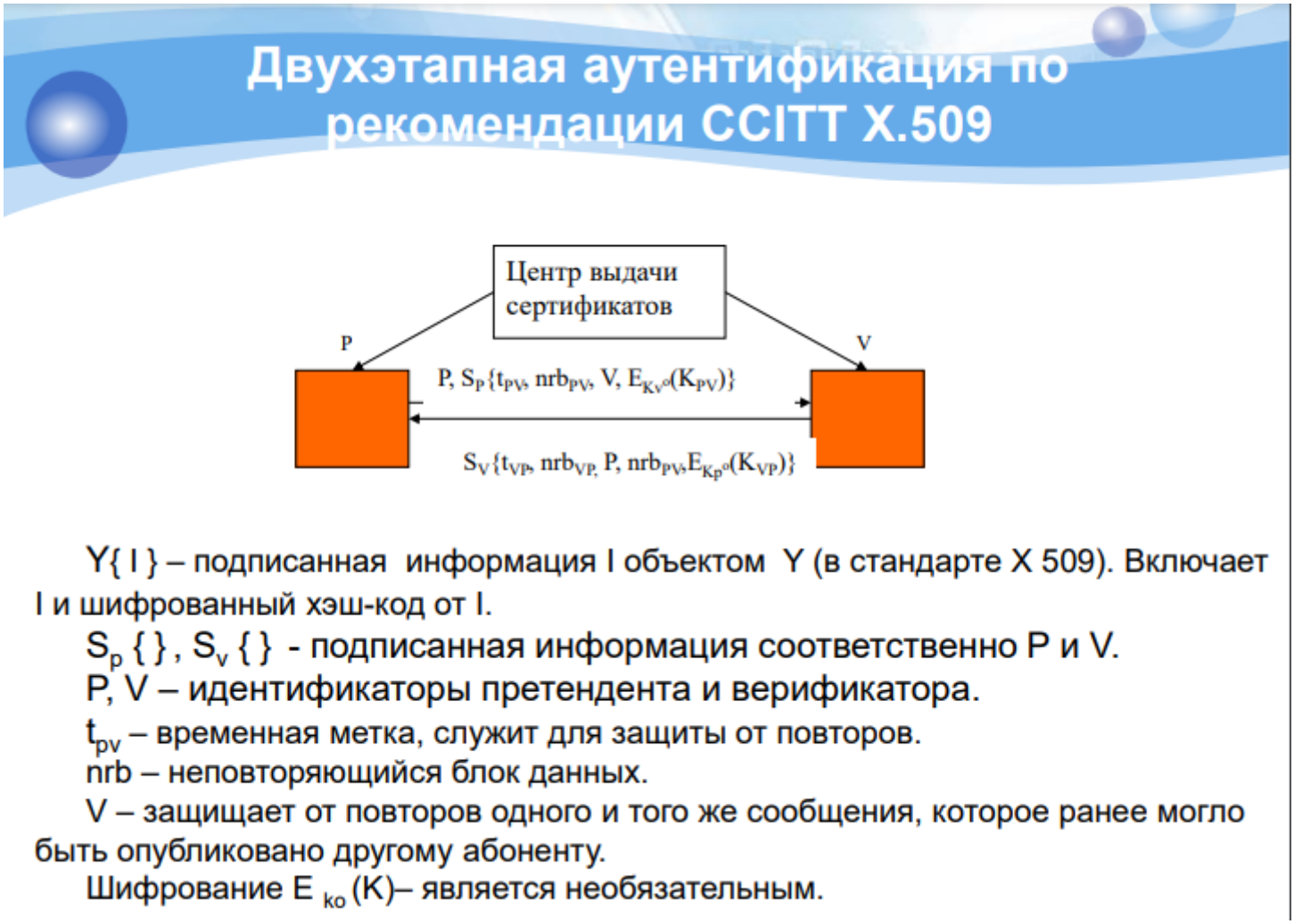
L – время жизни ключа

ПРОТОКОЛ СИСТЕМЫ KERBEROS

- Действия 1,2 выполняются один раз для каждого сеанса пользователя.
- Действия 3,4 выполняются один раз для каждого типа сервиса.
- Действия 5,6 выполняются один раз для каждого сеанса сервиса.
- Клиент использует мандат при запросах мандатов многократного доступа к службам.

49. Аутентификация субъекта в асимметричных системах по рекомендациям CCITT Recommendation X.509.

Двухэтапная аутентификация по рекомендации CCITT X.509



Y{ I } – подписанная информация I объектом Y (в стандарте X 509). Включает I и шифрованный хэш-код от I.

S_p { } , S_v { } - подписанная информация соответственно P и V.

P, V – идентификаторы претендента и верификатора.

t_{pv} – временная метка, служит для защиты от повторов.

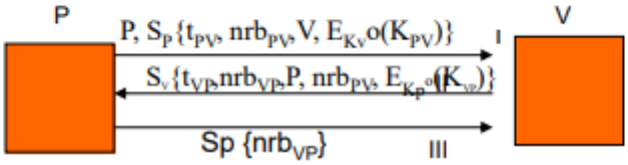
nrb – неповторяющийся блок данных.

V – защищает от повторов одного и того же сообщения, которое ранее могло быть опубликовано другому абоненту.

Шифрование E_{ko} (K)– является необязательным.

Трехэтапная аутентификация по рекомендации CCITT X.509

Трехэтапная аутентификация по рекомендации CCITT X.509

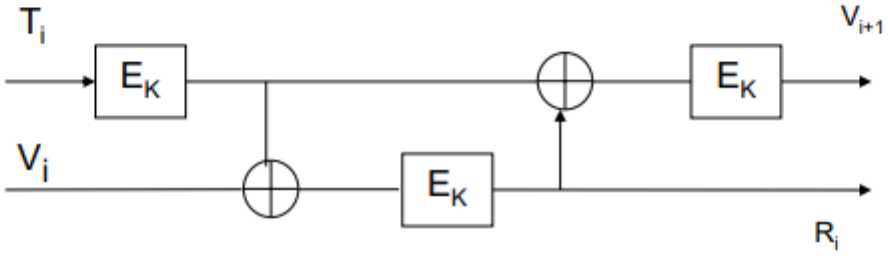


Оба **nrb** возвращаются назад другой стороной и может быть обнаружена атака воспроизведения.
Трехэтапная синхронизация **используется, когда синхронизация часов не возможна.**
Sp { } , Sv { } - подписанная информация соответственно P и V.
P, V – идентификаторы претендента и верификатора.
tpv,tvp – временные метки, служат для защиты от повторов.
nrb – неповторяющийся блок данных.
P – защищает от повторов одного и того же сообщения, которое ранее могло быть опубликовано другому абоненту.
Шифрование E ko (K) – является необязательным.

50. Генерация ключей по стандарту ANSI X 9.17.

Генерация ключей

В соответствии со стандартом ANSI X.917



$$R_i = E_K\,(E_K(T_i) \oplus V_i)$$
$$V_{i+1} = E_K\,(E_K(T_i) \oplus R_i)$$

К – ключ, зарезервированный для генерации;
V_0 – заготовка 64 битного секретного начального числа для генерации;
E_K — шифрование блочным шифром 64-разрядных блоков;
Т – текущее время с точностью до сотых долей секунды, введенное пользователем.

51. Хранение ключей согласно ISO 8532.

Стандарт ISO 8532 – устанавливает иерархию ключей, она может быть двух- и трехуровневой.

Двухуровневая:

- Ключи для шифрования ключей (КК)
- Ключи данных (ключи сеансовые, ключи рабочие) (КД)

Трехуровневая:

- Главные, мастер - ключи (ГК)
- КК
- КД

Кроме иерархии, стандартом устанавливаются и сроки хранения:

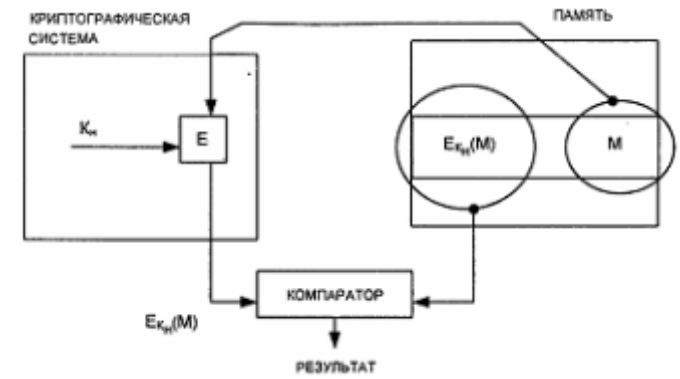
- ГК - доставляется при личном контакте, хранится от нескольких недель до месяцев в защищенной от сбоев и прочной криптосистеме.
- КД – в идеале должен меняться после каждого сеанса связи. Если ключи рабочие применяются для шифрования файлов, то срок действия может составлять несколько часов.

52. Мастер-ключ. Правила распространения и хранения.

На верхнем уровне иерархии ключей располагается главный ключ, мастер-ключ. Этот ключ применяют для шифрования КК, когда требуется сохранить их на диске. Обычно в каждом компьютере используется только один мастер-ключ. Мастер-ключ распространяется между участниками обмена неэлектронным способом - при личном контакте, чтобы исключить его перехват и/или компрометацию. Раскрытие противником значения мастер - ключа полностью уничтожает защиту компьютера.

Значение мастер - ключа фиксируется на длительное время (до нескольких недель или месяцев). Поэтому генерация и хранение мастер - ключей являются критическими вопросами криптографической защиты. На практике мастер-ключ компьютера создается истинно случайным выбором из всех возможных значений ключей. Мастер-ключ помещают в защищенный от считывания и записи и от механических воздействий блок криптографической системы таким образом, чтобы раскрыть значение этого ключа было невозможно. Однако все же должен существовать способ проверки, является ли значение ключа правильным. Проблема аутентификации мастер - ключа может быть решена различными путями.

Схема аутентификации мастер ключа



Администратор, получив новое значение мастер - ключа Кн хост-компьютера, шифрует некоторое сообщение М ключом Кн. Пара (криптограмма ЕКн(М), сообщение М) помещается в

память компьютера. Всякий раз, когда требуется аутентификация мастер-ключа хост-компьютера, берется сообщение М из памяти и подается в криптографическую систему. Получаемая криптограмма сравнивается с криптограммой, хранящейся в памяти. Если они совпадают, считается, что данный ключ является правильным.

53. Сеансовый ключ. Хранение.

Рабочие ключи (например, сеансовый) обычно создаются с помощью псевдослучайного генератора и могут храниться в незащищенном месте. Это возможно, поскольку такие ключи генерируются в форме соответствующих криптограмм, т.е. генератор ПСЧ выдает вместо ключа Ks его криптограмму EKн(KS), получаемую с помощью мастер-ключа хост-компьютера. Расшифровывание такой криптограммы выполняется только перед использованием ключа Ks.

Схема защиты рабочего (сеансового) ключа показана на рис.7.3. Чтобы зашифровать сообщение М ключом Ks, на соответствующие входы криптографической системы подается криптограмма EKн(KS) И сообщение М. Криптографическая система сначала восстанавливает ключ KS а затем шифрует сообщение М, используя открытую форму сеансового ключа Ks

Схема защиты ключа

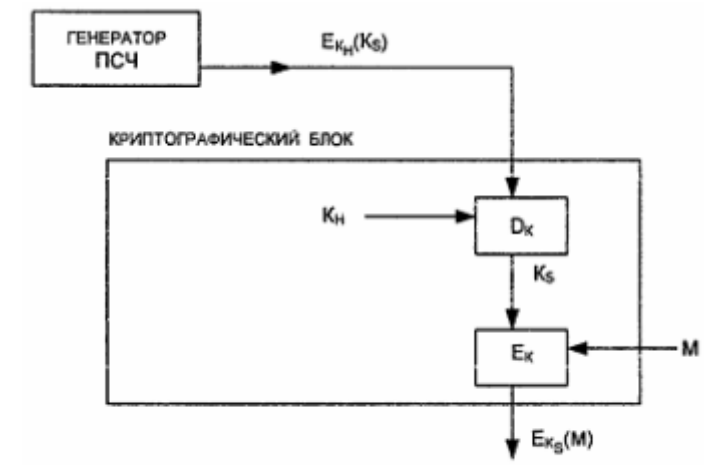


Рисунок 7.3 – Схема защиты ключа K_S

Таким образом, безопасность сеансовых ключей зависит от безопасности криптографической системы. Криптографический блок может быть спроектирован как единая СБИС и помещен в физически защищенное место.

Очень важным условием безопасности информации является периодическое обновление ключевой информации в АСОИ. При этом должны переназначаться как рабочие ключи, так и

мастер - ключи. В особо ответственных АСОИ обновление ключевой информации (сеансовых ключей) желательно делать ежедневно

54. Методы и средства контроля целостности сообщений. Использование шифрования, ЭЦП, кодов аутентификации сообщений, имитовставок.

Проверка целостности сообщения гарантирует **невозможность его модификации** из-за подмены злоумышленником или случайного искажения.

Защитить передаваемую информацию можно путём добавления к ней некоторого контрольного поля - **контрольной суммы.**
В зависимости от того, на основе какой информации и по каким правилам вырабатывается эта сумма, различают алгоритмы вычисления:

- хеш-функций
- имитовставки
- электронной цифровой подписи (ЭЦП).

ХЭШ

Хэш-функция – это преобразование, отображающее множество битовых строк произвольной длины на множество битовых строк фиксированной длины.
Криптографически стойкой называют хеш-функцию, удовлетворяющую следующим свойствам:

- вычислительно сложно найти текст, который при хешировании выдаст заданный хеш;
- вычислительно сложно найти коллизию – т.е. пару исходных текстов, которые дадут одинаковое значение хеша.

Под словосочетанием «вычислительно сложно» подразумевается, что для выполнения вычислений за разумное время необходима вычислительная мощность и/или объём оперативной памяти, превосходящий реально доступный на данном уровне развития вычислительной техники.

Недостатком использования хеш-функции для защиты информации является тот факт, что значение хеша зависит только от передаваемого сообщения. То есть, злоумышленник может изменить сообщение, и затем заново вычислить значение хеша, введя принимающую сторону в заблуждение

ИМИТОВСТАВКА

Для защиты от атак подмены сообщения и вычисления нового значения хеша применяются алгоритмы выработки имитовставки.
В англоязычной литературе такие алгоритмы называют MAC – Message Authenfication Code – код аутентификации сообщения.
Такие алгоритмы похожи на алгоритмы хеширования, с той разницей, что имитовставка зависит не только от исходного текста, но и от ключа.
Злоумышленник, не знающий ключа, не сможет рассчитать имитовставку для изменённого сообщения.
Таким образом, имитовставка может обеспечить все необходимые свойства сообщений – аутентификацию отправителя, целостность, невозможность отрицания авторства.

К недостаткам имитовставки можно отнести использование одного и того же ключа для выработки и для проверки имитовставки, что порождает проблемы распределения ключей, присущие всем методам симметричной криптографии.

ЭЦП

В алгоритмах ЭЦП используется не один ключ, а ключевая пара.
Один из ключей в этой паре называется открытым, второй – закрытым (секретным, личным).
Открытый ключ известен всем участникам, желающим получать сообщения от данного отправителя и проверять его подписи.
Закрытый ключ известен только отправителю сообщения.
Формирование ЭЦП осуществляется отправителем на основе сообщения и закрытого ключа.
Проверка ЭЦП осуществляется получателем с помощью сообщения, открытого ключа отправителя и цифровой подписи.
Таким образом, обеспечивается аутентификация, целостность и неотрицание авторства – сформировать верную подпись для сообщения может только отправитель, знающий закрытый ключ.

Такие системы позволяют обеспечить безопасную коммуникацию даже тогда, когда нет возможности распределить ключи по безопасному каналу. Однако схемы ЭЦП не свободны от недостатков.
Как правило, вычисление подписи требует больших затрат времени, чем формирование имитовставки.
Генерация ключевой пары занимает ещё больше времени.
Кроме того, необходимо контролировать передаваемые открытые ключи, чтобы злоумышленник не мог, перехватив их, организовать атаку «человек посередине».

55. Модели политики безопасности при построении защищенных систем. Дискреционное и мандатное управление доступом.

Политика безопасности - набор законов, правил и практических рекомендаций, на основе которых строится управление, защита и распределение критичной информации в системе.

Избирательная политика безопасности

Основой избирательной политики безопасности является избирательное управление доступом (ИУД, **Discretionary Access Control**; DAC), которое подразумевает, что:

- все субъекты и объекты системы должны быть идентифицированы;
- права доступа субъекта к объекту системы определяются на основании некоторого внешнего (по отношению к системе) правила (свойство избирательности).

Для описания свойств избирательного управления доступом применяется модель системы на основе матрицы доступа (МД, иногда ее называют матрицей контроля доступа). Такая модель получила название матричной. Матрица доступа представляет собой прямоугольную матрицу, в которой объекту системы соответствует строка, а субъекту - столбец. На пересечении столбца и строки матрицы указывается тип (типы) разрешенного доступа субъекта к объекту. Обычно выделяют такие типы доступа субъекта к объекту как "доступ на чтение", "доступ на запись", "доступ на исполнение" и др. Множество объектов и типов доступа к ним субъекта может изменяться в соответствии с некоторыми правилами, существующими в данной системе. Определение и изменение этих правил также является задачей ИУД. Например, доступ субъекта к конкретному объекту может быть разрешен только в определенные дни (дата - зависимое условие), часы (время - зависимое условие), в зависимости от других характеристик субъекта (контекстно-зависимое условие) или в зависимости от характера предыдущей работы. Такие условия на доступ к объектам обычно используются в СУБД. Кроме того, субъект с определенными полномочиями может передать их другому субъекту (если это не противоречит правилам политики безопасности).

Решение на доступ субъекта к объекту принимается в соответствии с типом доступа, указанным в соответствующей ячейке матрицы доступа. Обычно, избирательное управление доступом реализует принцип "что не разрешено, то запрещено", предполагающий явное разрешение доступа субъекта к объекту.

Вследствие больших размеров и разреженности МД хранение полной матрицы представляется нецелесообразным.

Избирательное управление доступом является основой требований к классам C2 и C1.

Полномочная политика безопасности

Основу полномочной политики безопасности составляет полномочное управление доступом (**Mandatory Access Control**; MAC), которое подразумевает что:

- все субъекты и объекты системы должны быть однозначно идентифицированы;
- каждому объекту системы присвоена метка критичности, определяющая ценность содержащейся в нем информации;
- каждому субъекту системы присвоен уровень прозрачности (security clearance), определяющий максимальное значение метки критичности объектов, к которым субъект имеет доступ.

В том случае, когда совокупность меток имеет одинаковые значения, говорят, что они принадлежат к одному уровню безопасности. Организация меток имеет иерархическую структуру и, таким образом, в системе можно реализовать иерархически ненисходящий (по ценности) поток информации (например, от рядовых исполнителей к руководству). Чем важнее объект или субъект, тем выше его метка критичности. Поэтому наиболее защищенными оказываются объекты с наиболее высокими значениями метки критичности.

Каждый субъект кроме уровня прозрачности имеет текущее значение уровня безопасности, которое может изменяться от некоторого минимального значения до значения его уровня прозрачности.

Для моделирования полномочного управления доступом используется модель Белла-Лападула (Bell-LaPadulla model), включающая в себя понятия безопасного (с точки зрения политики) состояния и перехода. Для принятия решения на разрешение доступа производится сравнение метки критичности объекта с уровнем прозрачности и текущим уровнем безопасности субъекта. Результат сравнения определяется двумя правилами: простым условием защиты (simple security condition) и -свойством (-property). В упрощенном виде, они определяют, что информация может передаваться только "наверх", то есть субъект может читать содержимое объекта, если его текущий уровень безопасности не ниже метки критичности объекта, и записывать в него, - если не выше (*-свойство). Простое условие защиты гласит, что любую операцию над объектом субъект может выполнять только в том случае, если его уровень прозрачности не ниже метки критичности объекта.

Полномочное управление доступом составляет основу требований к классу B1 , где оно используется совместно с избирательным управлением.

56. Алгоритм обработки битов защиты в UNIX.

Владелец		**Группа**			**Все пользователи**			
Чтение	Запись	Выполн.	Чтение	Запись	Выполн.	Чтение	Запись	Выполн.
1	2	3	4	5	6	7	8	9

Алгоритм предоставления прав доступа.

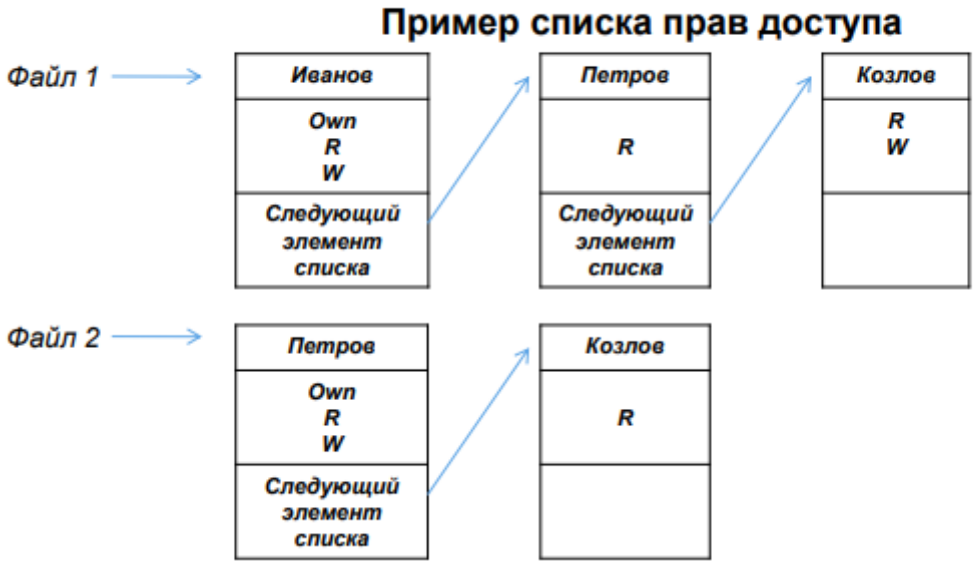
Субъект ассоциируется с эффективным идентификатором (EUID), содержащим информацию о пользователе и группе, к которой он принадлежит.

1. Проверяется, является ли субъект владельцем. При этом сравниваются значения EUID процесса и EUID' владельца объекта. Если EUID'= EUID, то сравниваются полномочия владельца с запрашиваемым типом доступа. Если запрашиваемый тип доступа присутствует в соответствующем поле, доступ предоставляется. Если нет, отклоняется. Если идентификаторы не равны, то осуществляется переход ко второму шагу.
2. Проверка соответствия полномочий входящего в группу владельца осуществляется аналогично п.1.
3. Сравниваются полномочия, предоставленные всем пользователям системы с запрашиваемым типом доступа. Если запрашиваемый тип присутствует в соответствующем поле, то доступ предоставляется, если нет – отклоняется.

57. Списки прав доступа ACL

ACL — определяет, кто или что может получать доступ к конкретному объекту, и какие именно операции разрешено или запрещено этому субъекту проводить над объектом.

Списки контроля доступа являются основой систем с избирательным управлением доступом.



Преимущество – возможность задания прав доступа индивидуально для каждого пользователя.

Недостаток – большие временные затраты на обработку списков по сравнению с доступом с помощью битов защиты.

В системе с моделью безопасности, основанной на ACL, когда субъект запрашивает выполнение операции над объектом, система сначала проверяет список разрешённых для этого субъекта операций, и только после этого даёт (или не даёт) доступ к запрошенному действию.

При централизованном хранении списков контроля доступа можно говорить о матрице доступа, в которой по осям размещены объекты и субъекты, а в ячейках — соответствующие права. Однако в большом количестве систем списки контроля доступа к объектам хранятся отдельно для каждого объекта, зачастую непосредственно с самим объектом.

Традиционные ACL системы назначают права индивидуальным пользователям, и со временем и ростом числа пользователей в системе списки доступа могут стать громоздкими. Вариантом решения этой проблемы является назначения прав группам пользователей, а не персонально.

58. Алгоритм обработки списков прав доступа (произвольное управление доступом) в системах WindowsNT.

Поддерживается произвольное управление доступом (discretionary access control, DAC) с помощью ACL. Каждый объект имеет свой ACL, который состоит из так называемых сущностей

контроля доступа (Access Control Entries – ACE). Существует 3 типа ACE: ACE Allowed (разрешает указанный тип доступа), ACE Denied (запрещает), третий используется для аудита. Первые 2 ACE содержат маску доступа, определяющую разреш. или запрещ. типы доступа.

Каждый список ACL содержит в себе набор записей контроля доступа (Access Control Entry, ACE). Каждая запись включает в себя следующие поля:

- Идентификатор безопасности (SID) пользователя или группы, к которым применяется данная запись;
- Маска доступа, определяющая набор разрешений на данный объект;
- Набор флагов, определяющих, могут ли дочерние объекты наследовать данную ACE;
- Тип ACE (разрешение, запрет или аудит).
  Если в дескрипторе безопасности отсутствует ACL, то объект считается незащищенным и получить к нему доступ могут все желающие. Если же ACL есть, но в нем отсутствуют ACE, то доступ к объекту закрыт для всех.

Алгоритм контроля доступа в системах Windows

1. Система сравнивает идентификатор процесса, запросившего доступ с идентификаторами, присутствующими в ACL объекта. Если в ACL отсутствует упоминание этого идентф. то доступ запрещается. Если идентиф. присутствует в ACL, то сначала обрабатывается ACE типа Denied, а затем типа Allowed.
2. Для всех ACE, имеющих тип Denied, запрашиваемый тип доступа сравнивается с указанным в маске ACE. Если хотя бы один тип доступа (чтение, запись и т.д.) присутствует в обеих масках, то доступ запрещается. После обработки всех ACE этого типа начинается обработка элементов типа Allowed.
3. Для всех ACE типа Allowed запрашиваемый тип доступа сравнивается с указанным в маске ACE. Если все типы доступа в запросе встретились в масках ACE, доступ разрешается, если не все – запрещается.

59. Мандатный доступ. Правила управления доступом

Мандатное управление доступом (Mandatory access control, MAC) — разграничение доступа субъектов к объектам, основанное на назначении метки конфиденциальности для информации, содержащейся в объектах, и выдаче официальных разрешений (допуска) субъектам на обращение к информации такого уровня конфиденциальности. Также иногда переводится как **Принудительный контроль доступа**.

При мандатном подходе политика доступа к информации задаётся не зависимо от пользователей системы и не может быть изменена в ходе работы системы.

В рамках этой модели безопасности фигурируют объекты (пассивные сущности) и субъекты (активные сущности): каждому объекту соответствует **уровень секретности** (например, знакомые любому слова "секретно" или
"совершенно секретно"), а субъекту - **уровень доступа**.

Свойства:

- задана упорядоченная последовательность уровней конфиденциальности (секретности) информации;
- каждому субъекту присвоен уровень доступа, определяющий к нему системы защиты информации.
- No read up (NRU) – нет чтения вверх: субъект имеет право читать только те документы, уровень безопасности которых не превышает его собственный уровень безопасности
- Субъект не может модифицировать объекты с более высоким уровнем доступа (нельзя писать вверх, No Write Down - NWD)

60. Свойства модели Белла-Лападула.

Модель Белла — Лападулы является моделью разграничения доступа к защищаемой информации. Она описывается конечным автоматом с допустимым набором состояний, в которых может находиться информационная система. Все элементы, входящие в состав информационной системы, разделены на две категории — субъекты и объекты. Каждому субъекту присваивается свой уровень доступа, соответств. степени конфиденциальности. Аналогично, объекту присваивается уровень секретности.

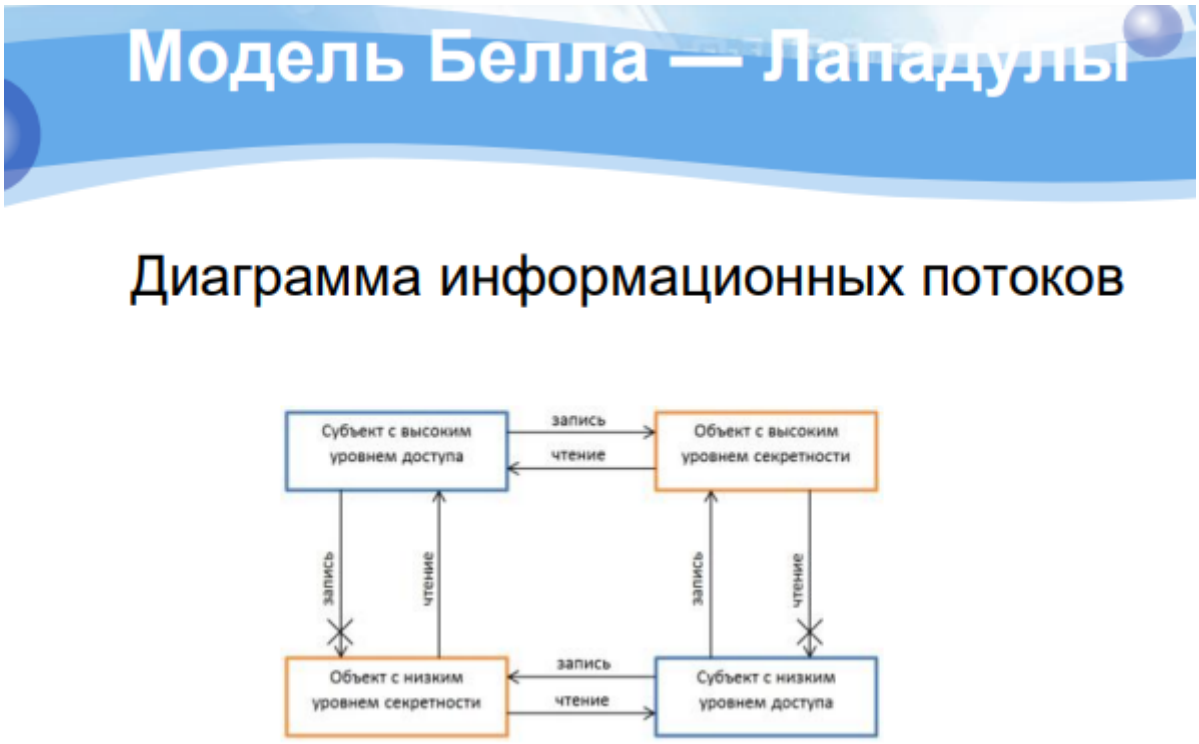
Понятие защищённой системы определяется следующим образом: каждое состояние системы должно соответствовать политике безопасности, установленной для информационной системы. Переход между состояниями описывается функциями перехода. Система находится в безопасном состоянии в том случае, если у каждого субъекта имеется доступ только к тем объектам, к которым разрешен доступ на основе текущей политики безопасности.

Модель Белла-Лападулы

- Простое свойство безопасности (запрет чтения с
  верхнего уровня).
  Субъект с уровнем безопасности X может читать

информацию из объекта с уровнем безопасности Y, только если X преобладает над Y.

- Свойство -* (star property) (запрет записи на нижний уровень).
  Субъект с уровнем безопасности X может писать информацию в объект с уровнем безопасности Y, только если Y преобладает над X.



61. Управление доступом в МСВС 3.0

Основные понятия

- Сопоставляются классификационные метки (метки безопасности) каждого субъекта и каждого объекта
- Метка субъекта описывает его благонадежность, метка объекта — степень закрытости содержащейся в нем информации
- Метки безопасности состоят из двух частей — уровня секретности и списка категорий.
- Уровни секретности образуют упорядоченное множество, которое может выглядеть, например, так: особо важно;совершенно секретно; секретно; конфиденциально; несекретно.
- Категории образуют неупорядоченный набор. Их назначение — описать предметную область, к которой относятся данные. В военном окружении каждая категория может

соответствовать, например, определенному виду вооружений.Механизм категорий позволяет разделить информацию по отсекам, что способствует лучшей защищенности.

Правила управления

- Управление доступом основано на сопоставлении меток безопасности субъекта и объекта
- Субъект может читать информацию из объекта, если уровень секретности субъекта не ниже, чем у объекта, а все категории, перечисленные в метке безопасности объекта, присутствуют в метке субъекта. В таком случае говорят, что метка субъекта доминирует над меткой объекта. Смысл сформулированного правила понятен — читать можно только то, что положено
- Субъект может записывать информацию в объект, если метка безопасности объекта доминирует над меткой субъекта. В частности, "конфиденциальный" субъект может писать в секретные файлы, но не может — в несекретные (разумеется, должны также выполняться ограничения на набор категорий).

Поддержка СВТ

- В СВТ реализован диспетчер доступа, т.е. средство, осуществляющее перехват всех обращений субъектов к
  объектам, а также разграничение доступа в соответствии с заданным принципом разграничения доступа. При этом
  решение о санкционированности запроса на доступ должно приниматься только при одновременном разрешении его и
  дискреционными, и мандатными ПРД.
- Поддерживается до 8 уровней секретности (от 0 — самый несекретный до 7 — самый секретный) и до 61 различных
  категорий.

Мандатные метки

1. Мандатные метки называются равными, если равны их уровни и равны наборы (векторы) категорий.
2. Мандатная метка М1 называется более высокой, чем
   мандатная метка М2, если:
   2.1. ее уровень выше (а векторы категорий равны)
   2.2 ее вектор категорий включает в себя вектор категорий метки М2 (а уровни равны)
   2.3 уровень М1 более высокий, чем уровень М2 и вектор категорий М1 включает в себя вектор категорий М2

3. Мандатные метки называются несравнимыми, если их векторы категорий не включают друг друга.

Доступ процессов к файловой системе
Согласно мандатной политике доступ процессов к файловой системе определяется следующим образом:

- если мандатные метки процесса и файла (каталога) совпадают (равны), то доступ разрешается полностью — так, как если бы мандатных ограничений не было;
- если мандатная метка процесса выше, чем мандатная метка файла (каталога), то доступ разрешается только на чтение и исполнение;
- если мандатная метка процесса ниже мандатной метки файла (каталога), то доступ разрешается только на запись;
- если мандатные метки процесса и файла несравнимы, то доступ запрещается полностью.

62. Аудит. Цели и этапы аудита безопасности.

Под аудитом понимается регистрация и учет всех происходящих в системе событий, в том числе все попытки идентификации и аутентификации пользователей, а также попытки доступа к объектам.

Аудит сам по себе не является средством защиты, так как непосредственно не противостоит угрозам безопасности, однако анализ протокола аудита позволяет:

- определять последствия тех или иных действий пользователей,
- прямо в процессе функционирования системы выявлять последовательности событий, указывающие на то, что система подвергается атаке со стороны нарушителей, и предпринимать необходимые действия для предотвращения таких атак.

Например, сделать так, чтобы после многократных некорректных попыток идентификации и аутентификации пользователя блокировалось устройство ввода/вывода, через которые производятся эти попытки.

Функции средств регистрации и учета

Разработка средств регистрации и учета событий включает в себя реализацию следующих четырех групп функций:

- отбор событий, подлежащих регистрации;

- регистрация и учет этих событий;
- анализ журнала событий и распознавание угроз;
- реакция на выявление угрозы.

Поля записей Trusted Mach

В Trusted Mach каждая запись в протоколе аудита содержит стандартный заголовок, состоящий из следующих полей:

- идентификатора события;
- идентификатора пользователя, инициировавшего событие;
- идентификатора группы пользователя, инициировавшего событие;
- уровня безопасности объекта, с которым связано событие;
- уровня безопасности пользователя, инициировавшего событие;
- терминала, с которым связано событие;
- реакции на событие (запись в журнал аудита или поднятие тревоги).

События для регистрации:

- Попытки идентификации и аутентификации (дата и время, результат попытки и, в случае успеха, полномочия, предоставленные пользователю).
- Попытки доступа к объектам (дата и время события, тип объекта, тип запрашиваемого доступа и результат запроса). Наиболее простой метод отслеживания доступа в микроядерной ОС – регистрация всех обращений к серверу имен. Дает полную картину того, какие предпринимались попытки осуществления доступа и насколько они были успешны.
- Создание и удаление объектов (дата и время события, тип объекта, тип операции (создание/удаление) и результат запроса).
- Модификация параметров системы, связанных с безопасностью. К этой группе событий относятся, во-первых, попытки пользователя модифицировать списки прав доступа, попытки изменения имен объектов, изменения параметров сеанса работы с системой. Кроме того, сюда же относятся попытки администратора остановить систему, изменить параметры аудита, добавить и удалить пользователей, изменить их атрибуты безопасности и т.д.

Протокол аудита

Доступ к файлам, содержащим протокол аудита, может контролироваться стандартными механизмами управления доступом. Администратор безопасности может также выбрать процедуру, которая должна быть инициирована, когда дисковое пространство исчерпано.

Система аудита должна позволять администратору безопасности задавать действия, которые должны быть автоматически предприняты в случае наступления того или иного события. Например, регистрация соответствующей записи в протоколе, посылка сообщения определенному пользователю или приостановка выполнения задач пользователя-нарушителя.