Лекция 9

Безопасность

- Разница надежности и безопасности
  - Надежность - случайные события
  - Безопасность - нарушения целостности, вызванные человеком

По Orange Book

Классы(уровни) безопасности систем

1. **D** - сюда попадают системы, которые при анализе не попали ни в один другой класс (зачастую уникальные случаи)
2. **C** (самый слабый класс безопасности) - наличие подсистем идентификации, аутентификации и авторизации (на основе дискреционного доступа) (Определения), Дискреционный доступ, Учет событий **(сначала протыкать все ссылки, а только потом переходить к следующим уровням)**
   1. **C1** = **C** + предполагает, что существует доверительная вычислительная база - вся процедура от аутентификации до авторизации централизованно контролируется
   2. **C2** = **C1** + требования грнулированности доступа по субъектам - в матрице один субъект больше не может представлять не одного пользователя (исключаем общий профиль, к примеру guest)
   - Класс **C** защищает от несанкционированного CRUD, но не защищает от того, что пользователь, который имеет доступ к данным, не сможет разгласить его тем, кто такого доступа не имеет
3. **B** - мандатное управление доступом
   - Есть перечень всех субъектов
   - Есть перечень всех объектов
   - Есть перечень меток(мандатов, уровней доступа) (упорядочены по строгости)
   - Каждому субъекту и каждому объекту ставится в соответсвие ровно одна метка
   - Любой субъект может получать доступ к объекту на запись, если уровень равен его или выше, и на чтение, если равен его или ниже
   - Получается, что я можем отправить информацию людям, у которых уровень доступа ниже

- И не можем переписать информацию более высокого уровня в более низкий, чтобы слить людям без доступа

1. **B1** = мы сами определяем, какие объекты будут иметь мандатный доступ
2. **B2** = все объекты и субъекты имеют метки
3. **B3** = выделенный админ безопасности, то есть отдельный админ по безопасности, который имеет доступ к контуру безопасности и остальные обычные админы

- Но, что если в компанию наймется разработчик, который оставит в коде бэкдор, который позволит ему украсть из системы много миллионов денег и при этом беспроблемно исчезнуть. Тогда может помочь система класса **A**

4. **A** = проверенный дизайн - не просто тестируется система, а контролируется абсолютно весь процесс ее создания. **Контроль обсолютно всех процессов.**

Определения

Проблема информационной безопасности - надо связывать физические лица и абстрактную сущность пользователя. Как это сделать?

Идентификация - присвоение субъекту идентификатора и фиксация этого в системе

Категории идентификации:

- То что субъект знает - пароль и прочая хуета
- То чем субъект владеет - смарт-карта, телефон субъетка и т.д.
- Неотъемлимая характеристика субъекта - биометрия(отпечаток пальца, фотография, радужка глаза и т.д.)

Аутентификация - сам факт предъявления идентификатора и его проверки (доказательство)

Авторизация - предоставление доступа

Дискреционный доступ

Строим "Дискреционную матрицу"

	S_1	S_2	S_3	...	S_n
O_1	R	RUD	RU	...	CRUD
O_2		CRUD		...	

	S_1	S_2	S_3	...	S_n
O_3	RU	W		...	CRUD
...	...	...	...	...	
O_k		CR	UD		CRUD

C - create, **R** - read, **U** - update, **D** - delete

Столбцы - субъекты
Строки - объекты (файлы, базы данных ...)
Пересечение - права доступа

Получаем для каждого субъета **вектор прав доступа**

- Часто, так как матрица разряженная, хранят лист для каждого отдельного объекта
- Аналогично можно хранить не в субъекте, а в объекта набор субъектов, которые имеют к нему доступ (решения принимается исходя из оптимизации по памяти)

А кто выдает эти права?

1. **Супервайзер** - единственный человек, который имеет право менять что-то в матрице
   - Плюсы: если что-то случилось, четко знаем, кто виноват
   - Минусы: не работает круглосуточно, если совершил пакость, то уже трудно контролировать
2. **Owner** - назначаем владельца каждому объекту, то есть делим ответсвенность
   - Плюсы: нельзя "единолично ахватить власть", разделение доступ
   - Минусы: могут кооперироваться, тяжелее контролировать
3. **Наследуемое право** - могу назначить любому субъекту права и он сможет давать их далее (передача прав по цепочке)
   - Плюсы: легко сделать много людей
   - Минусы: проблема с забиранием прав, неконтролируемое предоставление прав

Учет событий

Нужно все логировать

После Orange Book

Проеобразование дискреционной матрицы

Таблица ролей

	R_1	R_2	R_3	...	R_m
O_1	RUD	UD	R	...	
O_2	R		U	...	RU
O_3	RU	R	D	...	
...	...	...	...	...	U
O_k		R		W	CRUD

Таблица субъектов

	S_1	S_2	S_3	...	S_n
R_1	X		X	...	
R_2		X	X	...	
R_3	X			...	
...	...	...	...	...	
R_m					

- За эту **гибкость** мы платим **производительностью** (перемножение матриц)

Еще две составляющите безопасности:

- **Переход от учета к аудиту**
  - Появляются механизмы выявления аномалий (в отличии от учета)

    - **Аудит** - навешивание на журналы аналитических инструментов, пытаемся увидеть закономерность и нарущение закономерностей в каком-то параметре. **Пример:** 40 раз за 2 часа поменяти пароль))) Аномальное поведение подсказывает, что что-то тут не так
- **Внедрение шифрования** (чтобы нас не обокрали просто вытащив диск) - информация хранится не в явном виде
    - **Прозрачное шифрование БД**: данные шифруются при записи и дешифруются при чтении.
    - **Column level encryption** - отдельные столбцы дешифруются при чтении
    - **Шифрование файловой системы** - драйвер файловой системы перед чтением
      (то есть перед записью в файловую систему) шифрует данные (не спасаемся от админа, но спасаемся от остального персонала)
    - **Шифрование на уровне приложения** - храним ключи в самом приложении и данные шифруем на клиенте
        - Минусы: теперь индексы не работают, то есть **жертвуем производительностью**