**Лаба1**

**Политика безопасности** — это индивидуальный для каждой организации набор правил и средств для защиты информации.

Она реализует управление доступом через два основных вида:

1. **Избирательная (Дискреционная) политика**
   * **Суть:** Доступ определяет администратор, задавая, кто (субъект) и к каким данным (объектам) имеет право обращаться.
   * **Модель:** Проще всего представляется в виде **матрицы доступа** (строки — пользователи, столбцы — файлы, ячейки — права: чтение, запись и т.д.).
   * **Применение:** Широко используется в коммерческих организациях, так как гибкая и соответствует требованиям по разграничению прав.
2. **Полномочная (Мандатная) политика**
   * **Суть:** Доступ определяется **метками конфиденциальности** у данных и **уровнями допуска** у пользователей.
   * **Правило:** Пользователь может получить доступ только к тем данным, чья метка конфиденциальности не превышает его уровень допуска.
   * **Главная цель:** Жёстко предотвратить утечку информации «сверху вниз» (от более секретных данных к менее секретным пользователям).

**Ключевой вывод:** Выбор политики безопасности зависит от технологии и потребностей организации. Её реализация должна быть тщательно продумана, чтобы не заблокировать доступ к нужной информации, формально соблюдая правила.

**1. Основной принцип:**

* Каждый объект (файл, устройство) имеет **владельца** (пользователя).
* **Владелец** имеет полные права над своим объектом, включая право **передавать права** доступа другим пользователям.

**2. Механизм реализации:**

* Права доступа описываются с помощью **матрицы доступа**.
* **Строки** матрицы — это пользователи (U).
* **Столбцы** — это объекты (O).
* **Ячейка на пересечении** строки и столбца содержит **множество разрешённых действий** (S) данного пользователя с данным объектом (например, {Чтение, Запись}).

**3. Суть на примере:**

* В приведённом примере **Администратор** имеет полные права на все объекты.
* **Гость** может только читать "Файл\_1" и использовать CD-RW.
* **Пользователь\_1** является владельцем "Файла\_2" (имеет на него полные права) и может читать "Файл\_1".

**Главный вывод:** Дискреционная политика основана на воле владельца объекта, который сам решает, кому и какие права на него предоставить. Матрица доступа — это наглядное представление этой политики.

**Лаба2**

Подсистемы идентификации и аутентификации выполняют роль "переднего края обороны" в системах защиты информации, поскольку именно они первыми подвергаются атакам злоумышленников. Парольные системы, остающиеся наиболее распространенным методом аутентификации, требуют комплексного подхода к обеспечению безопасности.

К паролям предъявляются три основных требования: минимальная длина не менее 6 символов; использование различных групп символов (строчные и прописные буквы, цифры, специальные символы); запрет на использование реальных слов и имен собственных. К подсистеме аутентификации также выдвигаются требования: установка максимального срока действия паролей; ограничение количества попыток ввода (обычно не более трех); введение временной задержки при ошибочном вводе.

**Ключевой аспект:** Количественная оценка стойкости парольной защиты основана на вероятностной модели. Вероятность подбора пароля злоумышленником вычисляется по формуле P = (V·T)/Aᴸ, где V - скорость перебора, T - срок действия пароля, A - мощность алфавита, L - длина пароля. Для обеспечения заданного уровня безопасности необходимо определить минимальное количество возможных паролей S\* = [V·P/T] и подобрать такие параметры A и L, чтобы выполнялось условие S\* ≤ Aᴸ.

**Практическое применение:** Например, при вероятности подбора P=10⁻⁶, сроке действия пароля T=7 дней и скорости перебора V=10 паролей в минуту, требуемая стойкость обеспечивается паролем из 8 символов латинского алфавита (A=26) или 6 символов, включающих буквы и цифры (A=36). Это демонстрирует, как теоретические расчеты непосредственно влияют на практические рекомендации по составлению паролей.

Лаба3

**Простой анализ теории RSA**

**Основная идея:** RSA основан на том, что легко умножить два больших простых числа (p и q), но невероятно сложно разложить результат их умножения (n) обратно на множители. Это называется "задачей факторизации".

**Ключи:**

* **Открытый ключ (Public Key):** (e, n). Им шифруют сообщение. Его можно спокойно передавать всем.
* **Закрытый ключ (Private Key):** (d, n). Им расшифровывают сообщение. Его хранят в строжайшем секрете.

**Почему это работает?** Потому что зная e и n, практически невозможно вычислить d. Для этого нужно знать p и q, а факторизовать большое число n — очень долгая задача для современных компьютеров.

**Упрощенная методика работы с RSA (по шагам)**

**Шаг 1: Генерация ключей**

1. **Выбираем два простых числа:** p = 3, q = 11.
2. **Вычисляем модуль:** n = p \* q = 3 \* 11 = 33.
3. **Вычисляем функцию Эйлера:** φ(n) = (p-1)\*(q-1) = 2 \* 10 = 20.
4. **Выбираем открытую экспоненту**e**:** Это число должно быть взаимно простым с φ(n)=20 (т.е. их наибольший общий делитель, НОД, равен 1). Давайте возьмем e = 3. Проверяем: НОД(3, 20) = 1. Всё верно.
5. Применим алгоритм Евклида к числам 3 и 20:
   1. Найдем остаток от деления 20 на 3: 20mod  3=220mod3=2.
   2. Теперь нужно найти НОД(3, 2).
   3. Найдем остаток от деления 3 на 2: 3mod  2=13mod2=1.
   4. Теперь нужно найти НОД(2, 1).
   5. Найдем остаток от деления 2 на 1: 2mod  1=02mod1=0.
   6. Когда остаток становится 0, алгоритм завершается, и текущее значение делителя (1) является НОД.
6. Таким образом, НОД(3, 20) = 1.
7. **Вычисляем закрытую экспоненту**d**:** Это самое сложное. Нам нужно найти число d, которое удовлетворяет условию: (d \* e) mod φ(n) = 1. То есть (d \* 3) mod 20 = 1. Методом подбора или расширенным алгоритмом Евклида находим d = 7. Проверяем: 7 \* 3 = 21, 21 mod 20 = 1. Условие выполнено.

**Итог:**

* **Открытый ключ (e, n):** (3, 33)
* **Закрытый ключ (d, n):** (7, 33)

**Шаг 2: Шифрование**

Допустим, мы хотим зашифровать секретное число M = 7 (в реальности это код буквы или часть двоичного сообщения).

**Формула шифрования:** C = M^e mod n

* C — шифротекст (зашифрованное сообщение).
* M — исходное сообщение (число).

Подставляем наши значения: C = 7^3 mod 33

Считаем:

1. 7^3 = 343
2. 343 mod 33: Делим 343 на 33. 33 \* 10 = 330.
3. 343 - 330 = 13.

**Итог:** Мы зашифровали сообщение M=7 и получили шифротекст C=13. Передаем число 13 по открытому каналу.

**Шаг 3: Дешифрование**

Мы получили шифровку C=13. Чтобы прочитать её, используем закрытый ключ.

**Формула дешифрования:** M = C^d mod n

Подставляем наши значения: M = 13^7 mod 33

Считаем аккуратно, чтобы не получить огромных чисел:

1. 13^2 = 169. 169 mod 33 = 169 - (5\*33) = 169 - 165 = 4.
2. 13^4 = (13^2)^2 = 4^2 = 16. 16 mod 33 = 16.
3. 13^7 = 13^4 \* 13^2 \* 13^1 = 16 \* 4 \* 13 = 832.
4. 832 mod 33: Делим 832 на 33. 33 \* 25 = 825.
5. 832 - 825 = 7.

**Итог:** Мы расшифровали шифротекст C=13 и получили исходное сообщение M=7! Алгоритм сработал.

**Анализ вашего примера**

В вашем примере:

* p=7, q=17
* n=119
* φ(n)=96
* e=5
* d=77

**Шифрование M=19:**  
C = 19^5 mod 119 = 66 (вы правильно привели расчет).

**Дешифрование C=66:**  
M = 66^77 mod 119 = 19.

**Важный момент:** Возведение в огромную степень (например, 77) на практике выполняется не прямым умножением, а с помощью алгоритма быстрого возведения в степень по модулю, который не требует вычисления гигантских чисел.

**Выводы и важные замечания**

1. **Стойкость:** В реальных системах используются числа p и q длиной в сотни или тысячи бит. Число n получается настолько большим, что его факторизация займет миллиарды лет даже для суперкомпьютеров.
2. **Длина блока:** Как верно указано, сообщение разбивается на блоки, которые должны быть меньше n. Если блок окажется больше или равен n, шифрование и дешифрование будут работать некорректно.
3. **RSA на практике:** Сам по себе RSA медленный. Часто им шифруют не всё сообщение, а лишь случайно сгенерированный ключ для симметричного алгоритма (например, AES), который уже используется для шифрования основных данных.

Таким образом, вы освоили основную методику работы асимметричного шифрования RSA: генерацию ключевой пары, процесс шифрования с помощью открытого ключа и процесс дешифрования с помощью закрытого.

**Лаба 4**

**1. Цель работы**  
Изучить методы «привязки» программного обеспечения к конкретным характеристикам компьютера для повышения защиты от несанкционированного использования.

**2. Характеристики компьютера, используемые для привязки**

* Операционная система и её версия.
* Имя пользователя и имя компьютера.
* Наличие звуковой карты, принтера, сканера и других устройств.
* Дата создания BIOS.
* Серийный номер жесткого диска.
* Характеристики процессора (например, частота, количество ядер).

**3. Windows API**

* API (Application Programming Interface) — набор функций, предоставляемых ОС для работы приложений с системой и аппаратными ресурсами.
* Основные категории функций API:
  1. Работа с приложениями (запуск, закрытие, управление окнами).
  2. Графика (создание и вывод изображений).
  3. Системная информация (текущий диск, объем памяти, имя пользователя).
  4. Работа с реестром Windows (чтение, запись, модификация конфигураций).

**4. Реестр Windows**

* Центральная база данных конфигурации ОС, заменяющая устаревшие ini-файлы.
* Структура реестра включает несколько корневых разделов:
  + **HKEY\_CLASSES\_ROOT** — информация о типах файлов и OLE-операциях.
  + **HKEY\_CURRENT\_USER (HKCU)** — настройки оболочки текущего пользователя.
  + **HKEY\_LOCAL\_MACHINE (HKLM)** — информация о компьютере и драйверах.
  + **HKEY\_USERS (HKU)** — настройки оболочки для всех пользователей.
  + **HKEY\_CURRENT\_CONFIG** — конфигурация устройств Plug&Play.
  + **HKEY\_DYN\_DATA** — динамическая информация о состоянии оборудования.

**5. Особенности работы с реестром**

* Изменение реестра вручную может привести к сбоям ОС.
* Для получения информации о системе программы используют API-функции, а изменения в реестре обычно выполняются автоматически.