# Защита от копирования

1. **Какими требованиями должен обладать параметр к которому привязываешься?**
   1. УНИКАЛЬНОСТЬ (его нельзя подделать, например, имя учетной записи можно)
   2. НЕИЗМЕНЯЕМОСТЬ (он является важной частью компьютера, например, материнская плата, BIOS, HDD)
   3. ДОСТУПНОСТЬ
2. Что такое WMI (для винды), что такое DMI для ubuntu

# Энигма

1. **Для чего нужен рефлектор?** Ответ: Для того, чтобы процесс шифрования не отличался от процесса расшифровки. (Тут что-то про то, что на прямом ходу мы работаем со значениями по индексу, а на обратном с индексами по значению).
2. **Чем отличается расшифровка от зашифровки?** Ответ: ничем
3. **Что быстрее - прямой ход или обратный?** Ответ: прямой. Потому что на прямом ходе ты получаешь значение по индексу, а на обратном индекс по значению, что, очевидно, сложнее, т.к. подразумевает перебор всех значений.
4. **Как изменяется размер файла после шифрования?** Ответ: никак (тк шифруем посимвольно, один символ переводится ровно в один символ)
5. **Чем рефлектор отличается от ротора?**
6. **Процедура замены**

# AES

<https://habr.com/post/212235/> во если весь алгоритм нужен

1. **Как бороться с незаполненным до конца блоком?** Ответ: до конца блока заполнить 0
2. **Спрашивает про поле Галуа.** Ответ: числа в поле Галуа представляются как наборы коэффициентов перед многочленами. В нем переопределены операции сложения (xor) и умножения (Умножение на 1 - тождественно. умножение на 2 - это умножение на x, остальное выводится из этих двух.). Используем для того, чтобы не выйти за пределы 1-байтных чисел.
3. **Что будет если подать 1 байт?** Ответ: Дополняем 15 байт нулей
4. **Сколько раундов:** 10
5. **Раундовые ключи:** 11
6. **Откуда взялся 11 ключ:**
7. MixColumns перемешивает столбцы (1 арг сост, 2 матрица в поле Галуа, котороя состоит из байт, матрица констант, умножаем строку на столбец, это складывается в полу Галуа, каждый байт получившегося столбца из компонент, входящих в изначальный)
8. Зачем Галуа? Может получится 9-ый разряд.
9. Из чего состоит матричное умножение: умножение и сложение. И может не хватить.
10. Допустим умножаем на 2. Есть символ. В битовом представлении он должен быть меньше, чем …
11. Как это делается через полиномы?
12. ~~1 бит сдвигается, если меньше 80. И больше тоже??~~
13. В реальности все переводится в полином соответствующей степени и они перемножаются
14. Раундовых ключей 10. Первый нужен для
15. Почему с самого начала начинаем с раундового ключа?\* Это не случайно.

# DES

<https://2hourscrypto.info/> - есть видео лекции, в шапке ссылки на них, там подробнее объясняется вроде

<https://youtu.be/mE_s-R5wvpw?t=2343> -- лекция Яндекса

<http://www.enlight.ru/crypto/algorithms/des/des00.htm> -- еще хуйня какая-то

1. Как генерируются раундовые ключи?
2. Как зашифруется файл, если он меньше 64 бит?

# RSA

1. Знать как формируются ключи. От чего зависит открытый и закрытый ключ?
2. Как вы возводили в степень?
3. **Расширенный алгоритм Евклида** <http://fitp.ifmo.ru/shared/files/201111/1_278.pdf> 6 страница
4. Ну и алгоритм (лучше знать как именно в Вашем коде реализовано)
5. Нужно найти 2 ключа? Сначала по формуле Эйлера считаем фи. В промежутке ищем
6. Еще спрашивает что-то про взаимно простое число. Число Эйлера. Есть функция, проверяющая взаимно простое ли число е с фи. Остаток от деления. Не расширенный алг Евклида. Пока фи не равен нулю, считаем остаток от деления е на фи. Если оно простое, то получим 0. В конечном счете будет остаток от деления. 2-ое число результат.

# Подпись

1. **Как формируется подпись?** Прогнать документ через хэш-функцию, затем закодировать получившийся хэш приватным ключом RSA. Получившаяся подпись поставляется с документом, вместе с публичным ключом. При этом на публичный ключ имеется сертификат, что он принадлежит автору.

Алгоритмы хеширования знать не нужно, только названия  
SHA2, SHA3, MD6, мб еще что

1. **Как проверяется подпись?** Подпись расшифровывается с публичным ключом. Документ прогоняется через такую же хэш-функцию, как при зашифровки. Сравниваем расшифрованную подпись и хэш. Если совпадают - все ок.

# Хаффман

1. **Какими могут быть коды значений по размеру?** Если смотреть по дереву то от 1 до 255 байт (смотря от реализации мб сразу биты у вас конечно)
2. **Как хранится алфавит?** Либо деревом, либо таблицей частот
3. **И где хранится?** (ну тут у каждого свое)
4. **Как обрабатывается отступ до 8 бит, где хранится эта информация и как при расшифровке это работает?** Если максимальная длина 255, а была 8, то как файл может весить в итоге меньше - мы учитываем частоты, поэтому символы с коротким кодом компенсируют эту разницу.

# LZW

1. Что алгоритму надо знать для расшифровки?
2. Описать процесс расшифровки
3. **Что происходит, когда расшифровщик натыкается на символ из нижней части таблицы (>255)?** У него есть уже коды одинарных символов, так как он идёт по порядку, если код больше то он просто возьмёт сумму текущего и след. Вот [тут](https://neerc.ifmo.ru/wiki/index.php?title=%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_LZW) есть снизу пояснение. Там словарь заполняется одинарными каждый раз перед началом прогона алгоритма, если такого символа нет, значит это несколько символов то он берет предыдущий + текущий и заносит в словарь.

Блочные обрабатывают группами бит.

Поточные потоком. Для каждого потока разной длины. Не ограничены на длину. Обратная связь тоже существует. (На самом деле это одно и тоже).

Подстановки заменяются. Перестановки меняются местами.

**Какой алгоритм симметричный использовался в лабе. На СабБайтс. Используется подстановка.**