## Алгоритмы асимметричного шифрования

### Цель работы

Познакомиться с принципами работы протоколов рукопожатия в современных компьютерных системах

### Задания для выполнения

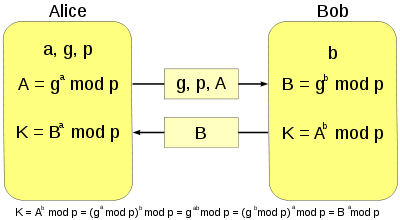
1. Реализовать протокол Диффи-Хеллмана в виде клиент-серверного приложения.
2. Реализовать клиент-серверную пару, которая шифрует сообщения асимметричным способом.

### Методические указания

#### Протокол Диффи-Хеллмана

Вашей задачей будет создать пару клиент-сервер, которые при подключении клиента к серверу реализуют установление общего секрета по протоколу Диффи-Хеллмана. Смысл этого протокола в том, чтобы вычислить общий секрет, то есть число, известное обоим сторонам общения без того, чтобы пересылать его по сети. Этот общий секрет может использоваться впоследствии для симметричного шифрования сообщений между этими сторонами.

Алгоритм основан на вычислении степеней числа по модулю. Для этого нам понадобятся два числа p и g. Они обычно берутся достаточно большими, чтобы взломать этот алгоритм было слишком сложно. Для целей обучения можно взять маленькие значения.



Сначала клиент генерирует секретное число a, затем вычисляет A = g^a mod p. Эти три числа он посылает на сервер. Сервер в свою очередь генерирует свое секретное число b и вычисляет B = g^b mod p. Его он отправляет на клиент. Сервер также берет число от клиента и вычисляет K = A^b mod p. Клиент же в свою очередь вычисляет K = B^a mod p. В итоге и на сервер и на клиенте должно быть вычислено одно и то же число K, которое никак не пересылалось и не может быть вычислено на основе той информации, что была передана по сети.

#### Асимметричное шифрование

Алгоритм Диффи-Хеллмана может быть также использован для асимметричного шифрования. В таком случае, набор (p, q, A) составляют открытый ключ клиента. Тогда вторая сторона может cгенерировать b, вычислить K и зашифровать любым симметричным шифром сообщение и послать его вместе с B. Тогда только данный клиент сможет расшифровать это сообщение, так как только он сможет вычислить из B правильное значение K, так как только он знает правильные значения p и q.

Ваша задача состоит в том, чтобы реализовать пару клиент-сервер, которые при подключении обменивались ключами и начинали общение в защищенном режиме.

Основной алгоритм работы клиента и сервера такой:

1. При запуске клиент и сервер генерируют каждый свою пару ключей.
2. При подключении клиент посылает серверу свой открытый ключ.
3. В ответ, сервер посылает клиенту открытый ключ сервера.
4. Клиент посылает сообщение серверу, шифруя его своим закрытым ключом и открытым ключом сервера.
5. Сервер принимает сообщение, расшифровывает его сначала своим закрытым ключом, а потом - открытым ключом клиента.
6. Обратное сообщение посылается аналогично.

### Контрольные вопросы

1. Как работает на практике протокол SSL?

### Дополнительные задания

1. Модифицируйте код клиента и сервера так, чтобы приватный и публичный ключ хранились в текстовых файлах на диске и, таким образом, переиспользовались между запусками.
2. Проведите рефакторинг кода клиента и сервера так, чтобы все, относящееся к генерации ключей, установлению режима шифрования, шифрованию исходящих и дешифрованию входящих сообщений было отделено от основного алгоритма обмена сообщениями.
3. (\*) Реализуйте на сервере проверку входящих сертификатов. На сервере должен храниться список разрешенных ключей. Когда клиент посылает на сервер свой публичный ключ, сервер ищет его среди разрешенных и, если такого не находит, разрывает соединение. Проверьте правильность работы не нескольких разных клиентах.
4. Модифицируйте код клиента и сервера таким образом, чтобы установление режима шифрования происходило при подключении на один порт, а основное общение - на другом порту. Номер порта можно передавать как первое зашифрованное сообщение.
5. (\*) Реализуйте пул портов.
6. Модифицируйте код FTP-сервера таким образом, чтобы он поддерживал шифрование.