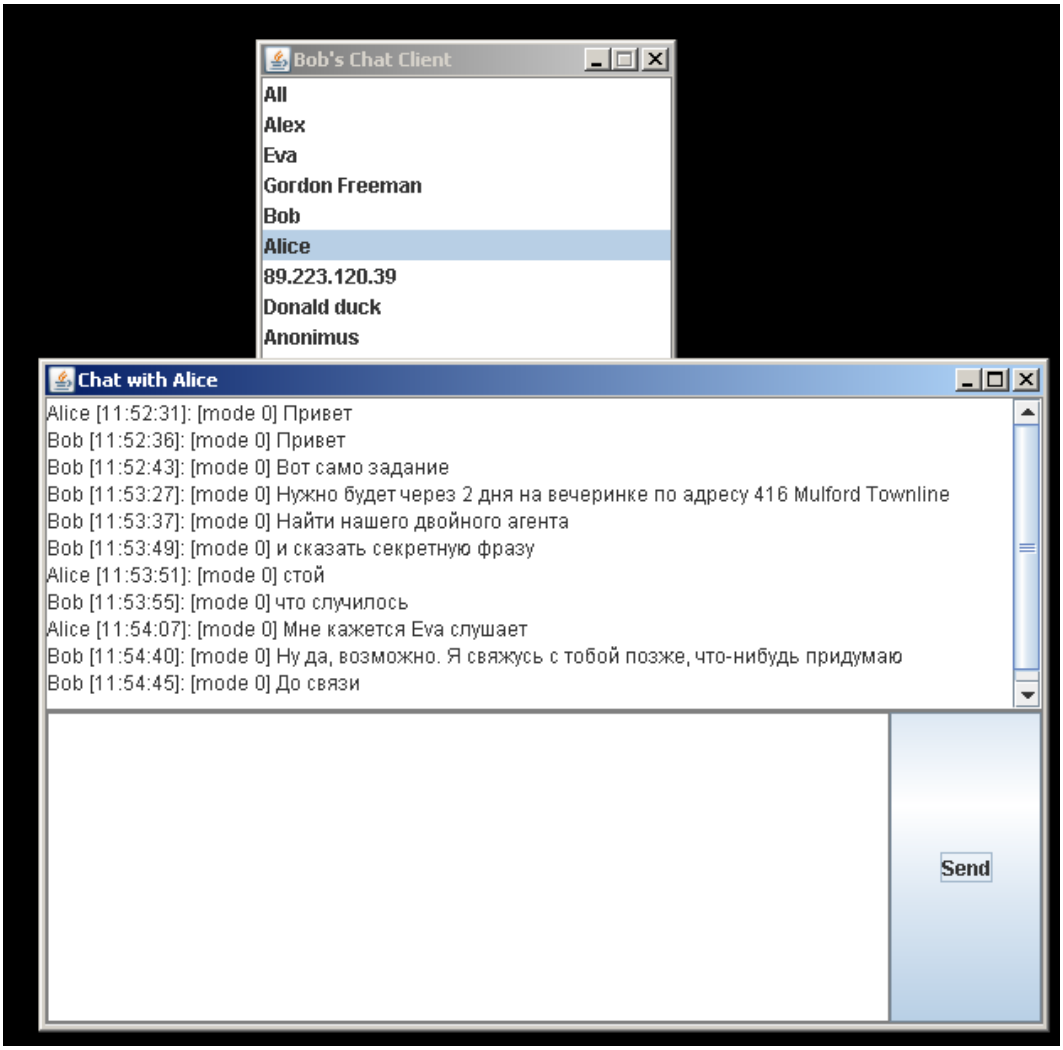
Проблема передачи сообщений

На протяжении всего времени, пока существует человечество, существовала проблема передачи сообщений так, чтобы его прочитать могли только отправитель и получатель. Всё это время велась борьба между шифровальщиками и крипто аналитиками: то первые одерживали вверх, то вторые. Но, как бы то ни было, крипто аналитики рано или поздно побеждали. Пока что всё иначе, во второй половине 20 века были придуманы алгоритмы ассиметричного шифрования Диффи-Хеллмана и RSA, которые произвели революцию в криптографии. Раньше, чтобы обменяться ключами нужно было сильно потрудиться и заплатить не мало денег. Нанимали специальных курьеров, которые доставляли ключи. И так для каждого получателя, не говоря уже о том, что через какое-то время ключи нужно менять и по пути могла произойти утечка этих ключей не в те руки. Сейчас же можно обмениваться данными по открытому каналу связи для всех и быть уверенным, что ваше сообщение не прочитают. Пока что... Как уже говорилось, исходя из истории, крипто аналитики всё равно побеждают. Или нет?

Представляю наших героев: Боб, Алиса и Ева. Боб и Алиса хотят обменяться секретной фразой, а Ева хочет узнать её. Все переписки будут происходить в написанной учебной программе, чтобы заодно с ней познакомиться. Начнём. Вот что видит Bob:



В общем проблема ясна. Нужно придумать способ, чтобы сказать секретную фразу. Боб нашёл алгоритм, позволяющий это сделать – это алгоритм Эль-Гамаля. Через некоторое время они снова вышли на связь.

Сначала Боб рассказал Алисе сам алгоритм.

Шаг 1: Боб генерирует числа P и A, такие, что P простое, а A является генератор мультипликативной группы кольца вычетов по модулю P.

Шаг 2: Боб генерирует числа X и Y - публичный и секретный ключи, такие что: $1 < X < P-1$, а $Y = A^X \bmod(P)$.

В данном случае всё это можно сделать командой: “!initEl”. Далее с помощью команд: “!ElGetP”, “!ElGetA” и “!ElGetY” можно получить числа P, A и публичный ключ Боба соответственно. Далее Боб отсылает Алисе эти данные.

Шаг 3: Алиса должна сгенерировать свои публичный и секретный ключи, зная P и A и отослать Бобу её публичный ключ.

Это можно сделать с помощь команды: “!initEl [P] [A]”, где [P] и [A] – это присланные Бобом числа P и A соответственно. Далее с помощью команды: “!ElGetY” можно узнать публичный ключ и отослать его Бобу. Затем с помощью команды: “!initContact [other Y]”, где [other Y] – это публичный ключ собеседника, можно проинициализировать публичный ключ собеседника.

Шаг 4: теперь любое сообщение m Алиса может зашифровать следующий образом:

Алиса выбирает взаимно простое с (P-1) число k.

Алиса вычисляет числа $r = A^k \bmod(P)$ и $e = (m*Y_b^k) \bmod(P)$, где Y_b – публичный ключ Боба.

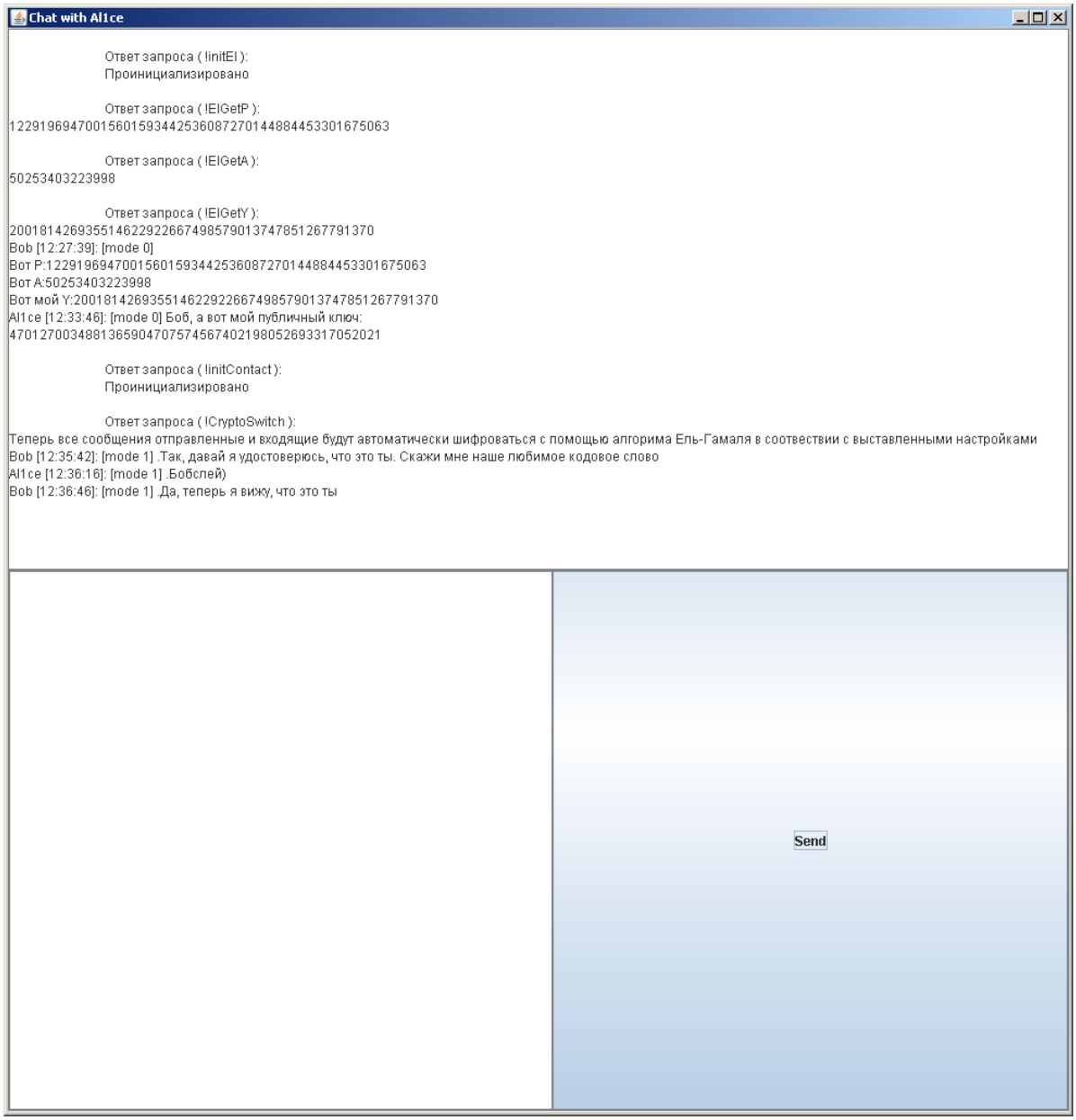
Теперь Алиса отсылает числа (r; e) Бобу.

Шаг 5: Боб дешифрует сообщение $m = (e*r^{(P-1-Xb)}) \bmod(P)$, где X_b – секретный ключ Боба.

Боб таким же образом может отправлять сообщения Алисе.

В программе после выполнения описанный команд выше можно включить шифрование автоматически командой: “!CryptoSwitch ElGamal”.

Вот как это всё выглядело у Боба:



Команды Боба:

!initEl

!ElGetP

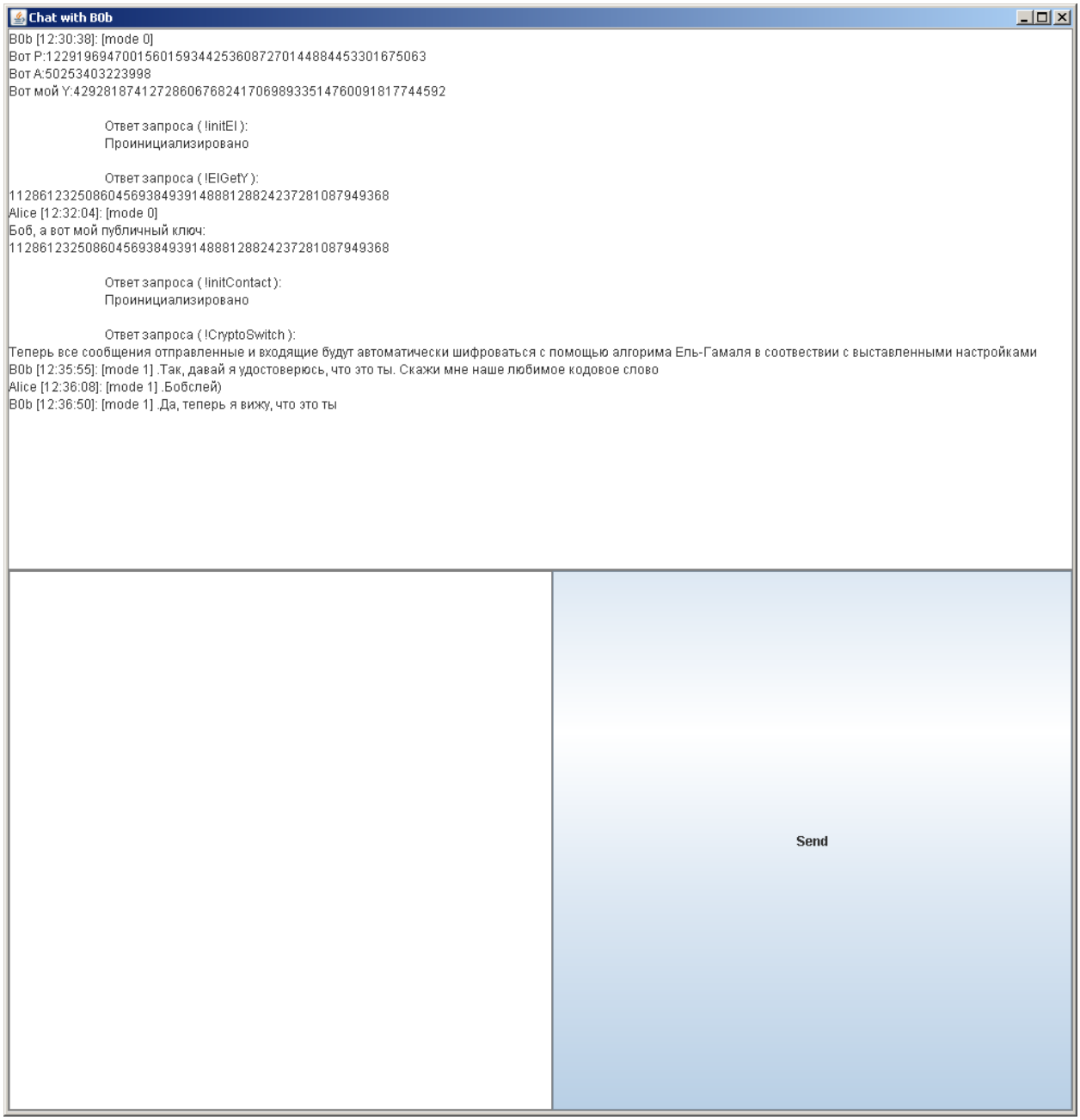
!ElGetA

!ElGetY (и отослал P, A и Y Алисе)

!initContact 470127003488136590470757456740219805269331705 2021

!CryptoSwitch ElGamal

И у Алисы:



Команды Алисы:

!initEl 122919694700156015934425360872701448844533016750 63

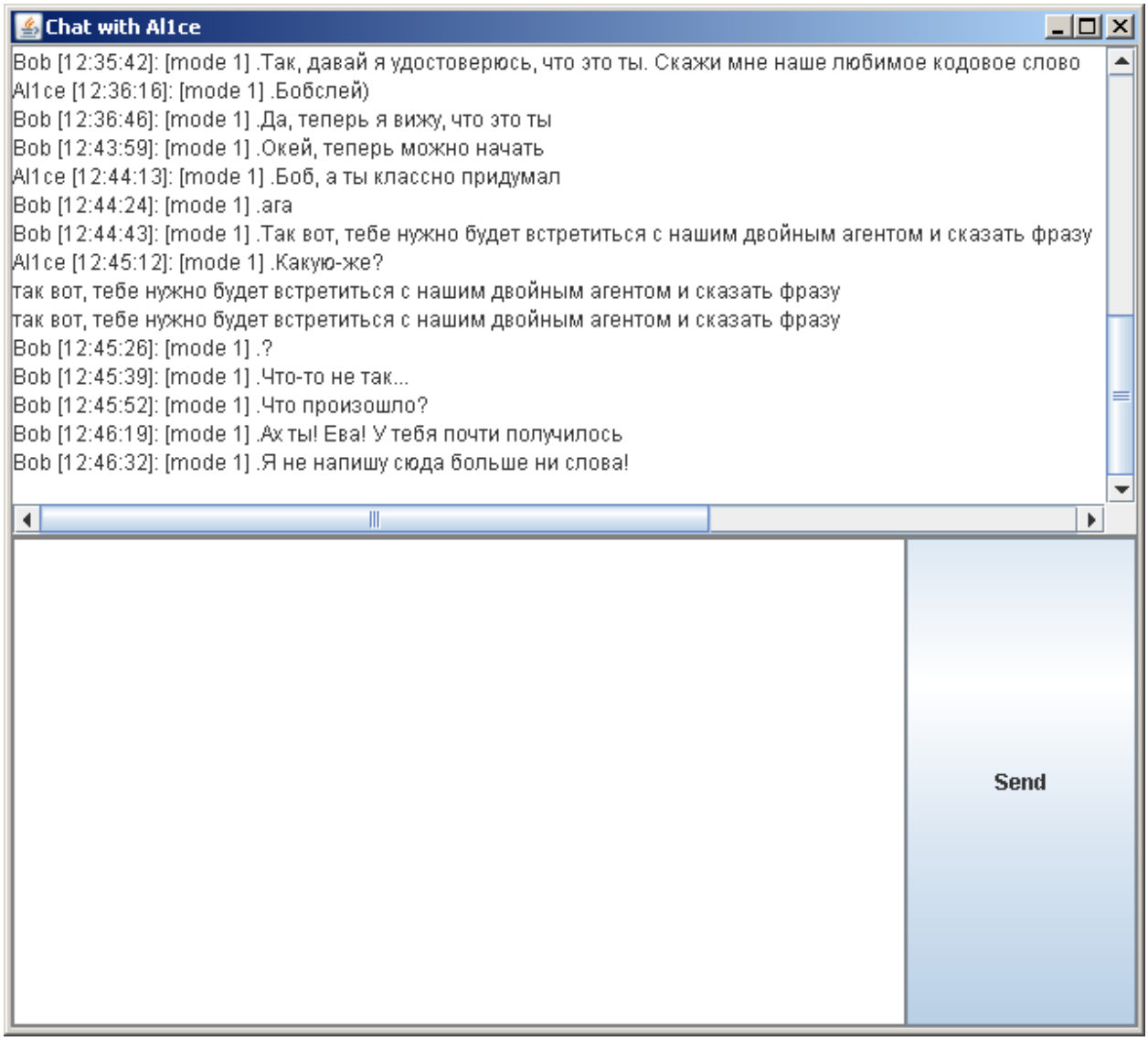
50253403223998

!ElGetY (и отослала его Бобу)

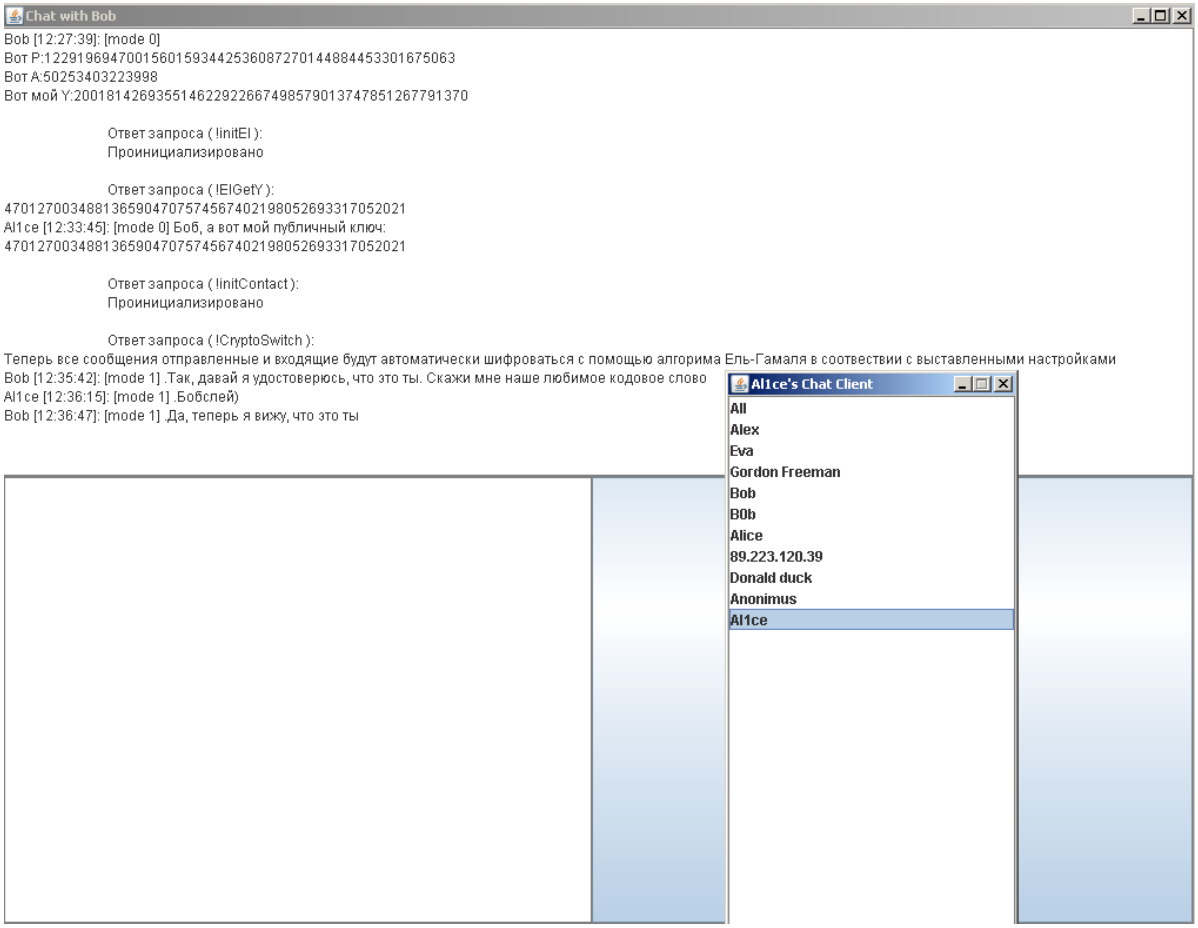
!initContact 42928187412728606768241706989335147600918177445 92

!CryptoSwitch ElGamal

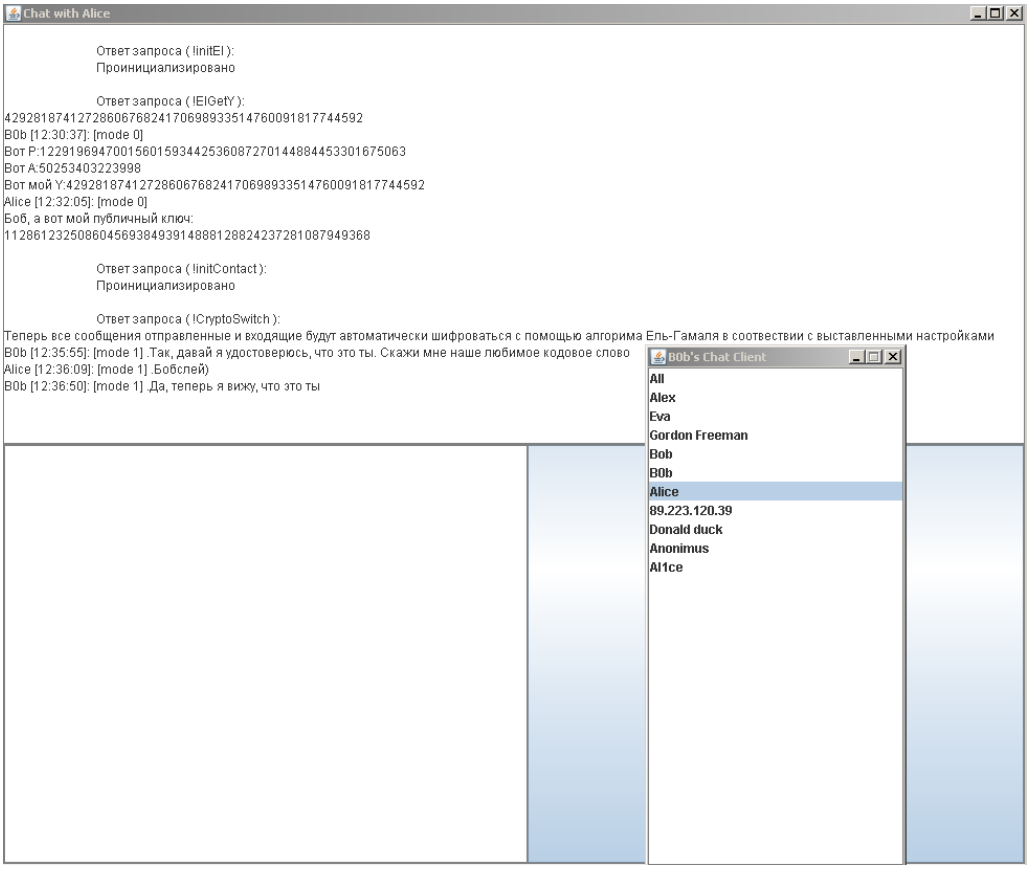
Теперь они могут писать, а все сообщения будут автоматически шифроваться с помощью алгоритма Эль-Гамаля, об этом можно узнать по стоящему рядом с сообщениями “[mode 1]”:



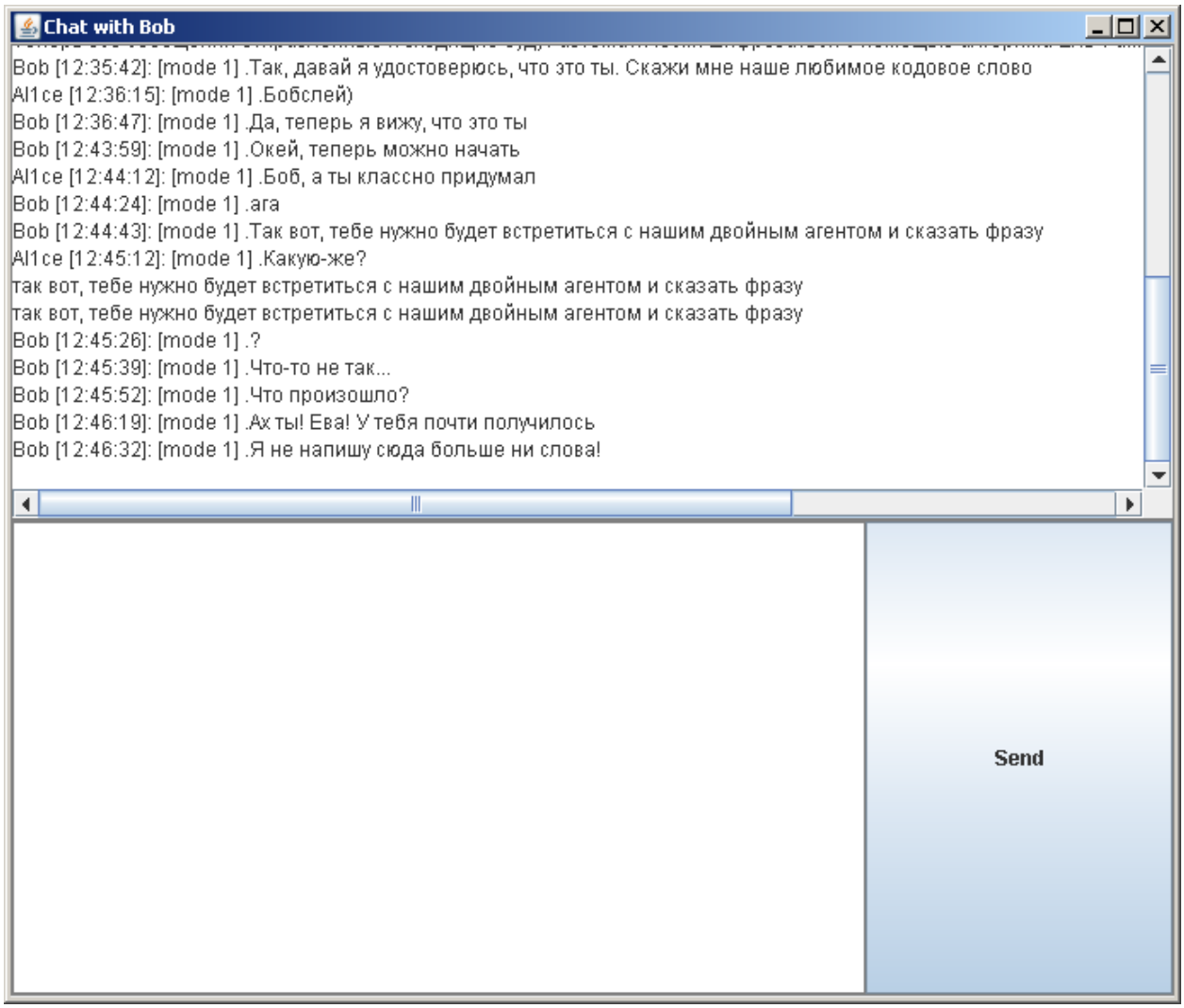
Что же произошло? Как Боб понял, что тут замешана Ева? Всё просто. Он увидел с кем общается: “Al1ce”. В имени вместо “i” стоит цифра 1 (тоже самое и с “B0b”, тут вместо “o” цифра 0). А вот что всё это время видела Ева при общении с Бобом:



При общении с Алисой:



Ева даже не знала эту кодовую фразу “Бобслей”. Она подменила публичные ключи, а Боб и Алиса даже этого не заметили, более того, Боб чуть не сказал секретную фразу Еве. Ева была тем самым “человеком посередине” и все сообщения шли через неё, она просто копировала сообщения из одного окна в другое и нечаянно вставила 2 раза скопированную фразу, ну что ж – повезло Бобу. Вот чат Евы с Бобом:



Как же решить проблему? Тут в игру вступает Peter_TheKeyKeeper (или Пётр). Это человек, которому все доверяют, он подписывает ключи, Пётр рассказал всем свой публичный ключ и параметры P и A сначала по радио, а потом выложил на многих форумах. Прежде чем подписать чей-то ключ, он удостоверяется в личности этого человека. В нашем случае Алисе повезло, и она знает этого человека лично. Peter_TheKeyKeeper с удовольствием подпишет ключ Алисы.

Как же подписать ключ?

Шаг 1: Пётр получает публичный ключ Алисы и добавляет к нему описание. Таким образом получается M.

Шаг 2: Peter_TheKeyKeeper вычисляет с помощью хеш-функции f(x) число m = f(M).

Шаг 3: Peter_TheKeyKeeper вычисляет числа k, r и e:

$1 < k < P-1$ и k взаимно простое с P-1; $r = A^k \bmod(P)$; $e = ((m - X_p*r)*k^{-1}) \bmod(P-1)$, где X_p – это публичный ключ Петра.

Peter_TheKeyKeeper отсылает Алисе M и (r, e).

В данному случае Петру нужно просто воспользоваться командой “SignKey [key] [msg]”, где [key] – это публичный ключ Алисы, а [msg] – это описание к этому ключу, пусть этим описанием будет “AliceKey”. Программа вычислит одно число, которое нужно отправить Алисе.

Шаг 4: теперь любой может проверить этот ключ. Если выполняется условие: $(Y_p^r * r^e) \bmod(P) = A^m \bmod(P)$ - то подпись верна (придётся снова вычислить m = f(M)). Y_p – публичный ключ Peter_TheKeyKeeper.

В программе можно выполнить команду: “VerifyKey [key] [other Y]”, где [key] – подписанный ключ Петром (то число, которое вычислила программа у Петра), а [other Y] – это публичный ключ Peter_TheKeyKeeper.

Расскажу про ещё 2 команды: можно сразу зашифровать сообщение без инициализации шифровальщика “!ElEn [P] [A] [other Y] [SrcMsg]” и можно сразу расшифровывать “!ElDe [P] [A] [X] [CryptoMsg]”, где [P] и [A] параметры P и A соответственно, [other Y] – публичный ключ того, кому собираемся отправлять сообщение, [SrcMsg] – сообщение, которое мы хотим зашифровать, [X] – секретный ключ того, кто расшифровывает сообщение, и [CryptoMsg] – зашифрованное сообщение.

И так имеем параметры Peter_TheKeyKeeper:

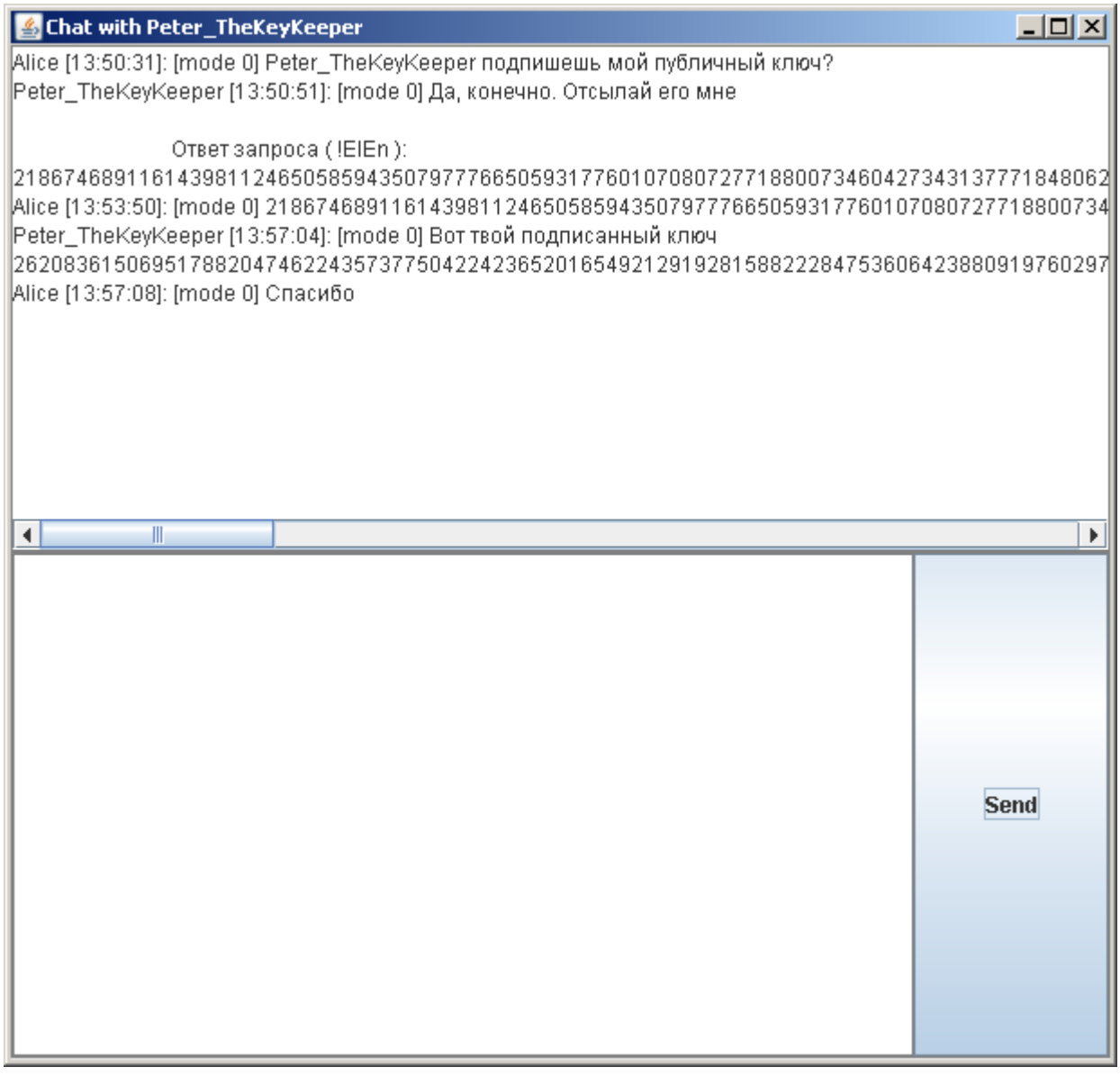
Публичный ключ Y_p:

181767117656539137451660990715950199664675019756 10,

P: 125718069992393723156400155887962930317402814441779 – обозначим *P*,

A: 71832929668563227852101920237208 – обозначим это *A*.

Вот что делала Алиса:



Сначала Алиса выполнила команду

!initEl *P A* или

!initEl 125718069992393723156400155887962930317402814441779718329296685632278521019202372 08

Далее она узнала свой публичный ключ с помощью команды “!ElGetY”:

267772291907208110290439337011137881092250715055 07 – обозначим Y_a

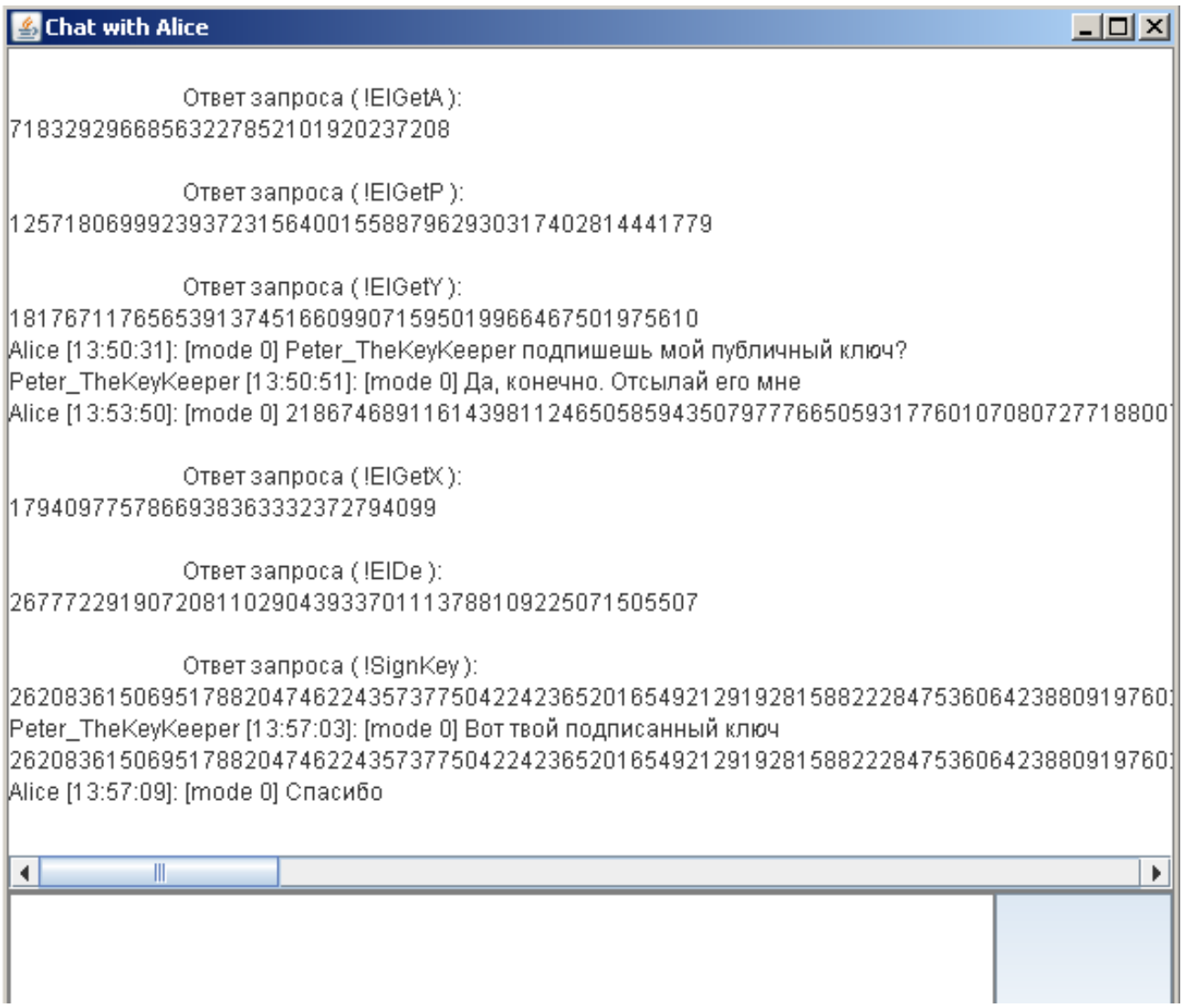
Алиса зашифровала свой публичный ключ и отослала Петру:

!ElEn P A Y_p Y_a или

!ElEn 125718069992393723156400155887962930317402814441779718329296685632278521019202372 08 181767117656539137451660990715950199664675019756102677722919072081102904393370111378810922507150550 7

и получила зашифрованное сообщение: 218674689116143981124650585943… - обозначим $CryptoY_a$, которое она отправила Петру:

Что делал Пётр:



Пётр проверил, что все его параметры верные: “!ElGetA”, “!ElGetP”, “!ElGetY”

Затем он узнал свой секретный ключ с помощью “!ElGetX”:

179409775786693836333237279 4099 – обозначим X_p

Далее он расшифровал сообщение, которое ему прислали:

!ElDe P A X_p $CryptoY_a$ или

!ElDe 125718069992393723156400155887962930317402814441779 718329296685632278521019202372 08 718329296685632278521019202372 08 179409775786693836333237279 4099 2186746891161439811246505859…

Тем самым получил публичный ключ Алисы. После он подписал её ключ:

!SignKey Y_a *“AliceKey”* или

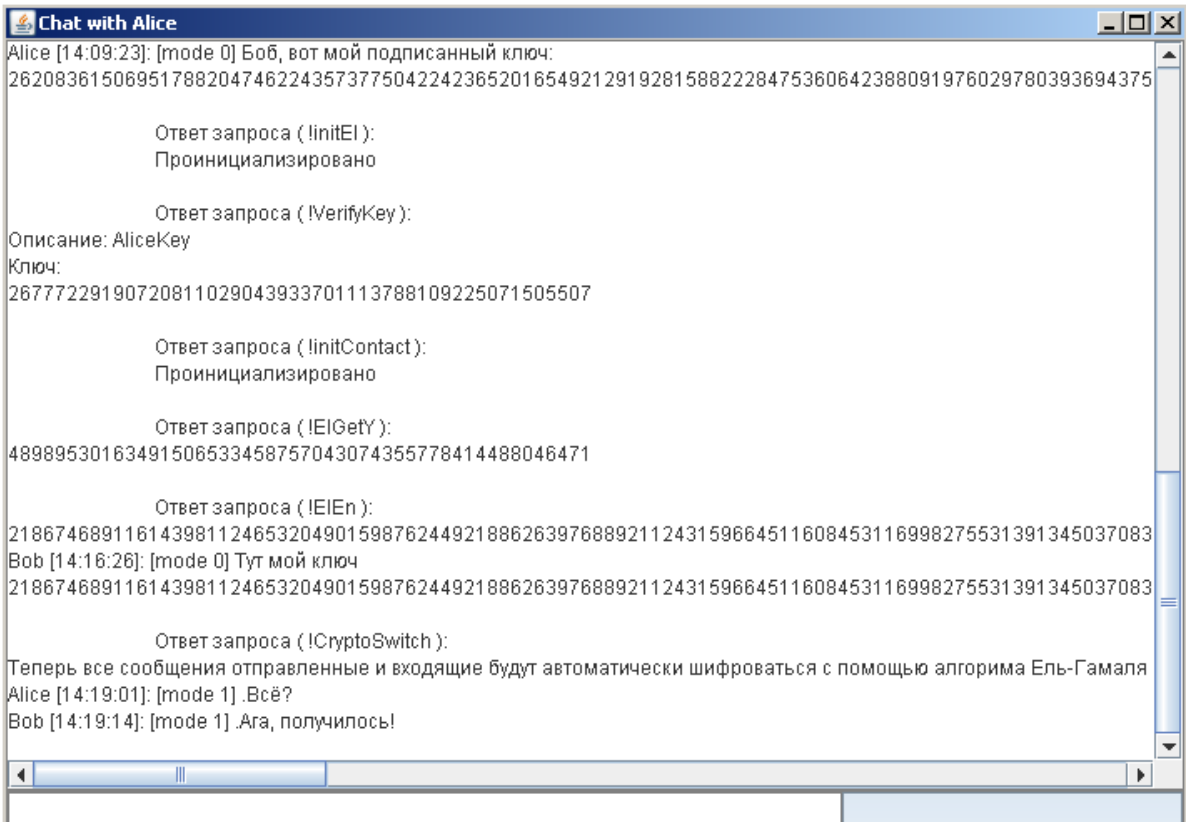
!SignKey 267772291907208110290439337011137881092250715055O7 AliceKey

И получил подписанный им ключ Алисы

26208361506951788204746224357377504224236520165492129192815882228475360642388091976029780393694375… - обозначим $SignedY_a$

В данной ситуации Ева уже ничего не может подменить, так как публичный ключ Петра все знают. А если она попробует, то после проверки подписанного ключа станет видно, что его подменили.

Теперь уже общаются Алиса и Боб. Вот что делал Боб:



Боб проинициализировал шифровальщик:

!initEl *P A* или

!initEl 125718069992393723156400155887962930317402814441779

718329296685632278521019202372O8

И проверил подпись (на этом шаге стало бы известно, что подпись фейковая или сам подписанный ключ был подменён)

!VerifyKey $SignedY_a$ Y_p или

!VerifyKey 26208361506951788204746224357377 5…
181767117656539137451660990715950199664675019756 10

Боб узнал свой публичный ключ с помощью “!ElGetY”:

489895301634915065334587570430743557784144880464 71 – обозначим Y_b

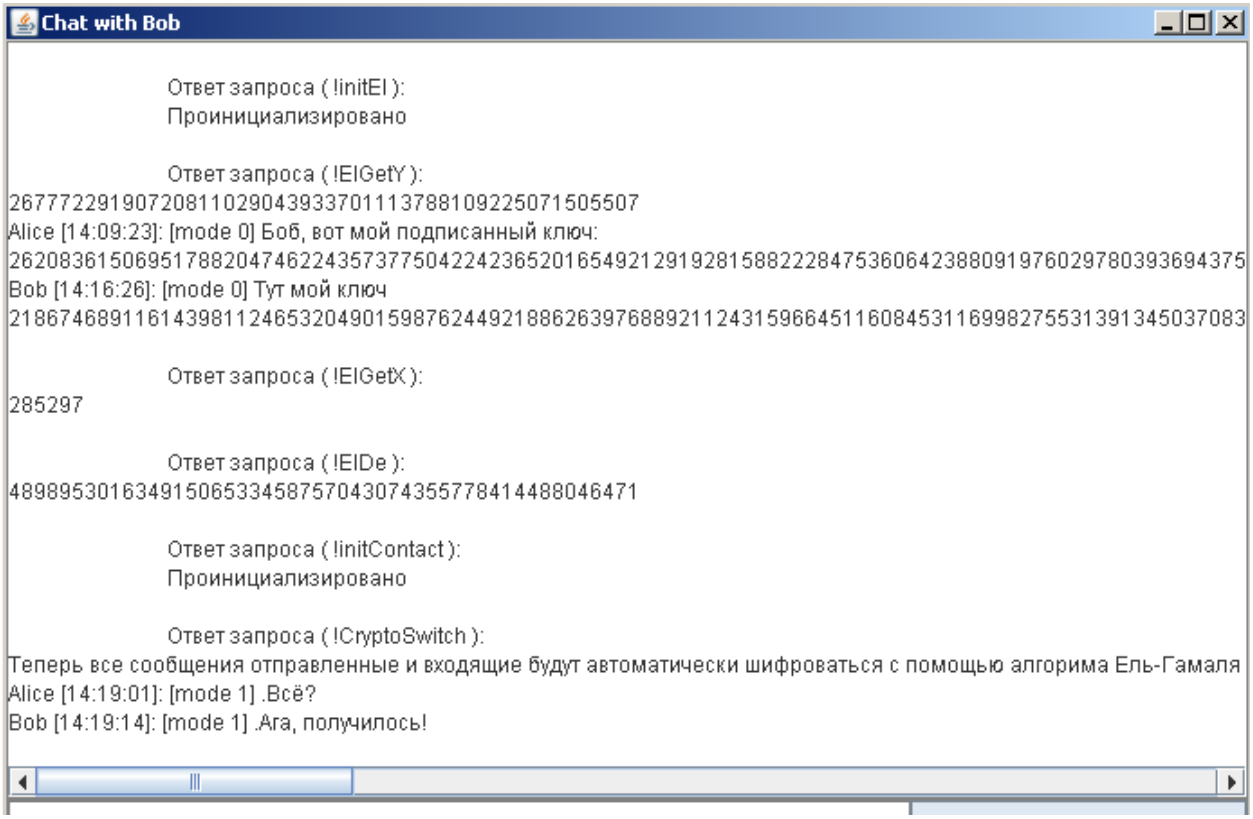
Далее он зашифровал его для Алисы:
!ElEn P A Y_a Y_b или

!ElEn 125718069992393723156400155887962930317402814441779
718329296685632278521019202372 08
267772291907208110290439337011137881092250715055 07
489895301634915065334587570430743557784144880464 71

и получил свой зашифрованный ключ для Алисы:

218674689116143981124653204901598762449218862… - обозначим $CryptoY_b$

Теперь что делала Алиса:



Она узнала свой секретный ключ с помощью команды “!ElGetX”:

285297 – обозначим X_a

Далее расшифровала сообщение от Боба:

!ElDe P A X_a $CryptoY_b$ или

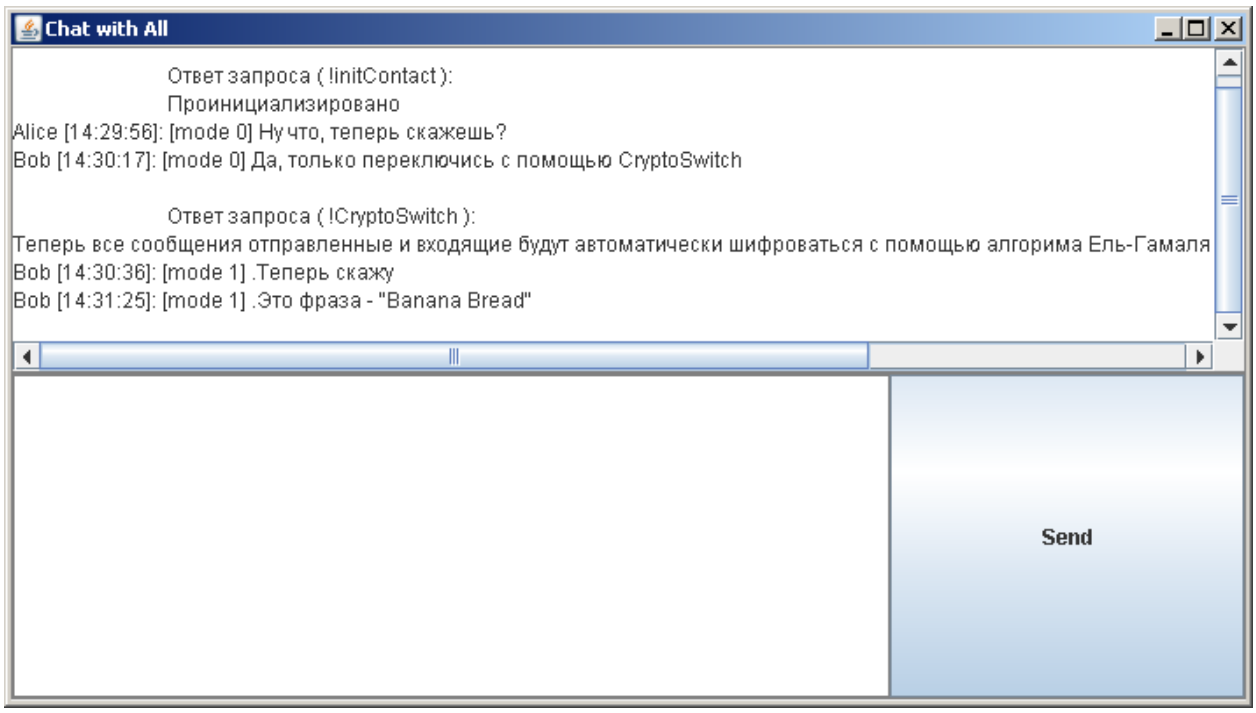
!ElDe 125718069992393723156400155887962930317402814441779
718329296685632278521019202372O8 285297
21867468911614398112465320490159876244921886 2…

И тем самым получила публичный ключ Боба Y_b:

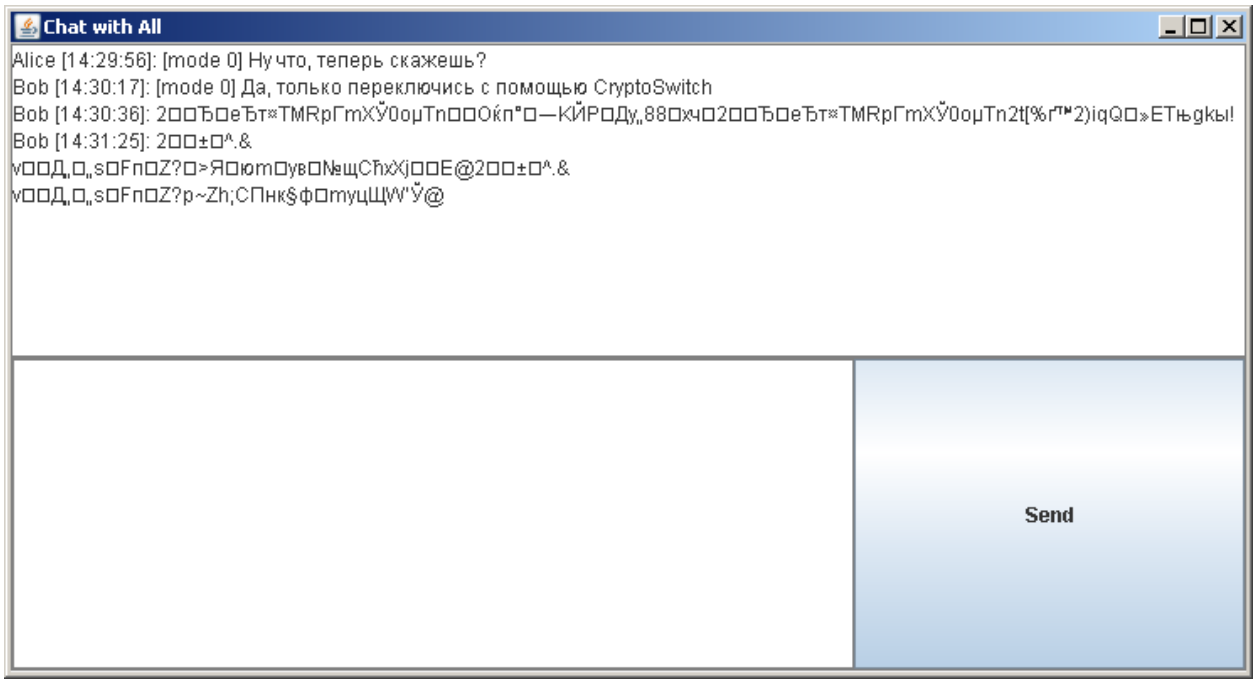
48989530163491506533458757043074355778414488046471

Если Ева на данном шаге будет менять сообщения, то об этом узнают Боб и Алиса, получив неверные публичные ключи.

Теперь наконец узнаем это кодовое слово. Алиса и Боб перешли в общий чат, чтобы подразнить Еву. Вот что видела Алиса в общем чате.



Вот что видела Ева в общем чате.



Вот так Алиса и Боб смогли поделиться кодовой фразой “у всех на виду”. Вся криптостойкость этого алгоритма основана на сложности вычисления дискретных логарифмов. Кто знает, может скоро крипто аналитики снова одержат вверх, когда найдут простой способ вычислять дискретные логарифмы с помощью квантовых компьютеров или как-то ещё. Или же гонка закончилась?