Формальная верификация криптографических протоколов с использованием Proverif

Винарский Евгений

Институт системного программирования

28 июля 2021

Слабые места защиты информации

- Атаки на архитектуру (Криптографическая система не может быть надежнее использованных в ней отдельных алгоритмов шифрования)
- Атаки на конкретные реализации
  - Переполнение буферов
  - Не стёртая до конца секретная информация
- Атаки на сетевое оборудование
- Атаки на пользователей
- Атаки с использованием побочных каналов
- ...

Для того чтобы преодолеть систему защиты, достаточно взломать любой из ее компонентов

Криптографические примитивы

- Алгоритмы симметричного шифрования
- Алгоритмы ассимметричной криптографии (выработка общих сессионных ключей и т.д.)
- Датчик псевдослучайных чисел
- Алгоритмы хэширования

В этом блоке считаем, что все криптографические примитивы не могут быть скомпрометированы раньше, чем перестанут использоваться

Какие достоинства и недостатки асимметричной криптографии?

Модель противника

- **Модель атаки** Возможности противника по взаимодействию с системой
- **Ресурсы противника** Предположения о вычислительных и информационных ресурсах противника
- **Угроза** Задача противника по нарушению свойств безопасности

Уязвимости криптосистемы возникают, если неправильно выбраны

1. модель атаки
2. угроза
3. предположения о ресурсах

Модель противника (2)

- *Пассивный противник* (противник может читать зашифрованные пересылаемые данные в открытом канале)
- ***Dolev-Yao (Активный) Противник*** может:
  - читать сообщения в канале
  - модифицировать сообщения в канале
  - удалять сообщения из канала
- Противник, учитывающий временные задержки (может определить, какая именно проверка не прошла, . . .)
- . . .

Рассуждать о стойкости криптосистемы можно только в терминах модели противника

Протокол (алгоритм) Диффи-Хеллмана

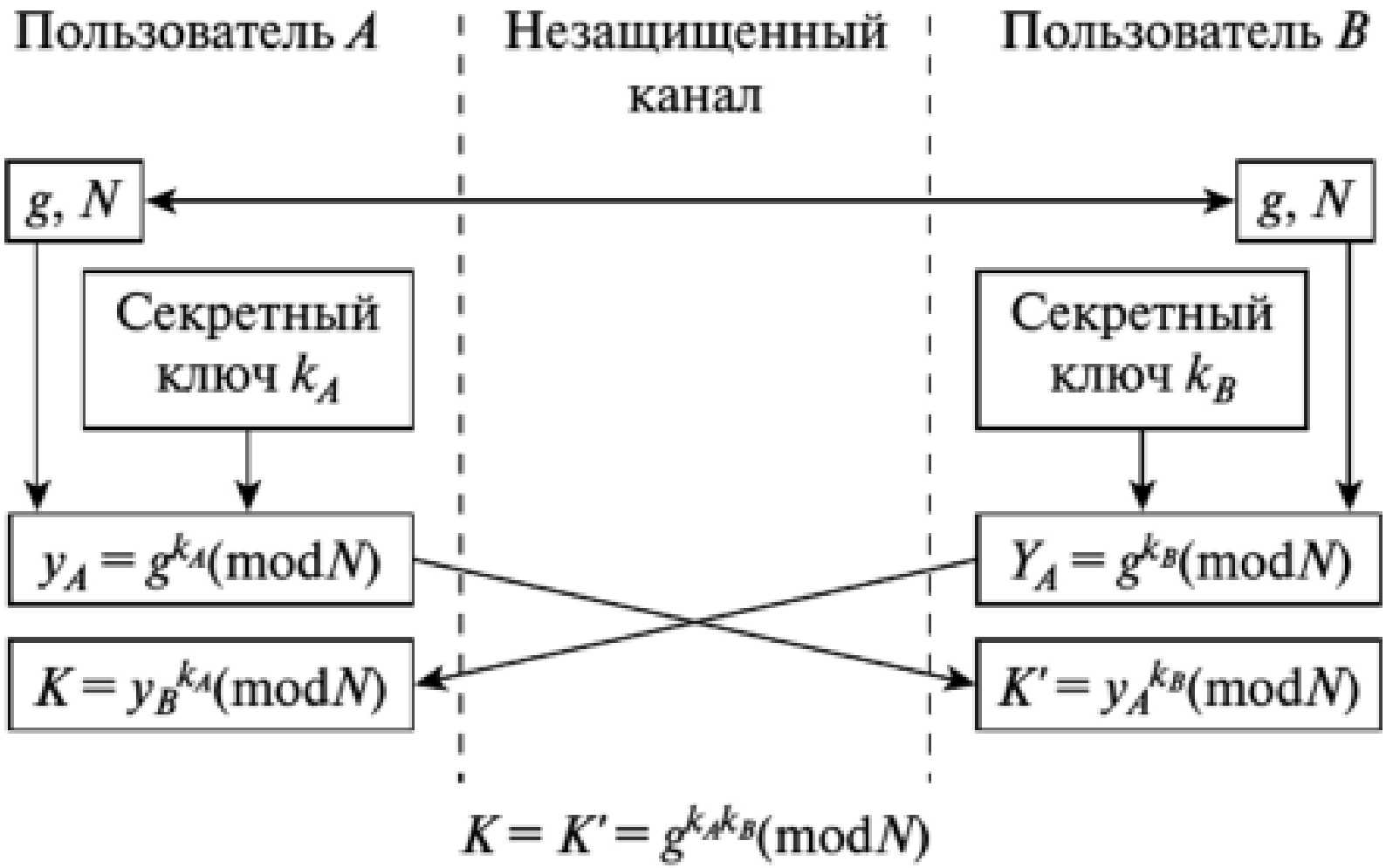


Схема протокола Диффи-Хеллмана

Модель противника

Протокол Диффи-Хелммана при активном противнике

- E – пассивный противник, слушающий незащищённый канал
- E известно значение g, N и y_A, y_B

Угроза: Противник E узнал выработанный общий ключ $K = g^{k_A k_B} \ (mod \ N)$

Модель противника

Протокол Диффи-Хелммана при активном противнике

- E – пассивный противник, слушающий незащищённый канал
- E известно значение g, N и y_A, y_B

Угроза: Противник E узнал выработанный общий ключ
$K = g^{k_A k_B} \ (mod \ N)$
Пусть E скомпрометировал $K = g^{k_A k_B} \ (mod \ N)$, тогда:

- узнал k_A
  - решил задачу дискретного логарифмирования, т.е. вычислил k_A из уравненя $y_A = g^{k_A} \ (mod \ N)$
- узнал k_B
  - решил задачу дискретного логарифмирования, т.е. вычислил k_B из уравненя $y_B = g^{k_B} \ (mod \ N)$
- узнал $k_A * k_B$

Протокол Диффи-Хэллмана стойкий по отношении к пассивному противнику

Свойства безопасности протоколов выработки общих ключей обмена

- Аутентификация
  - Ложная аутентификация
  - Unknown key share (Неизвестный общий ключ)
- Установление одинаковых ключей
- Секретность ключей обмена
- Уникальность установленных ключей обмена
- Forward secrecy (Прямая секретность)
- Backward secrecy (Обратная секретность)

Протокол Нидхема-Шрёдера

Протокол выработки общего сессионного ключа (N_A, N_B)

Alice (pkA, privA)		Bob (pkB, privB)
- генерация N_A - шифрует $\langle A, N_A \rangle_{pk(B)}$	$\langle A, N_A \rangle_{pk(B)}$ ---------->	
	$\langle N_A, N_B \rangle_{pk(A)}$ <----------	- дешифровка сообщения $\langle A, N_A \rangle_{pk(B)}$ - $\langle N_A, N_B \rangle_{pk(A)}$
- дешифровка сообщения $\langle N_A, N_B \rangle_{pk(A)}$ - $\langle N_B \rangle_{pk(B)}$	$\langle N_B \rangle_{pk(B)}$ ---------->	
(N_A, N_B) -- общий секрет		

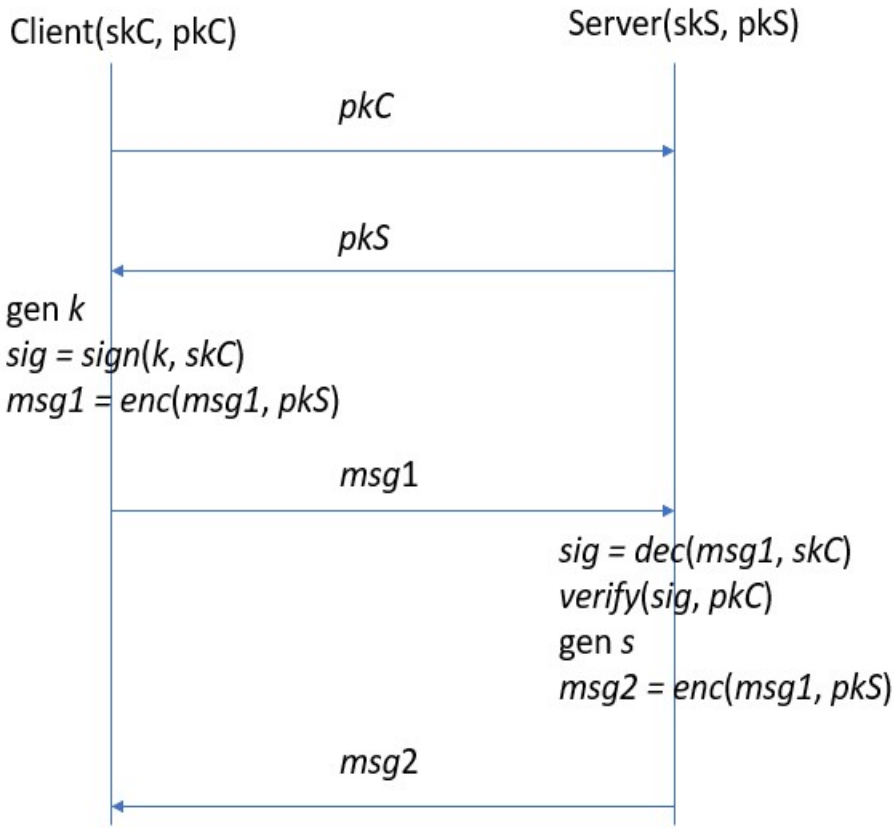
Уязвимость протокола Нидхема-Шрёдера

Alice	Intruder (pkI, privI)	Bob
- генерация N_A - шифрует $\langle A, N_A \rangle_{pk(I)}$ $\langle A, N_A \rangle_{pk(I)}$ ---------->		
	- дешифрует $\langle A, N_A \rangle_{pk(I)}$ - шифрует $\langle A, N_A \rangle_{pk(B)}$ $\langle A, N_A \rangle_{pk(B)}$ ---------->	
		- дешифровка сообщения $\langle A, N_A \rangle_{pk(B)}$ - $\langle N_A, N_B \rangle_{pk(A)}$ $\langle N_A, N_B \rangle_{pk(A)}$ <----------

Уязвимость протокола Нидхема-Шрёдера (2)

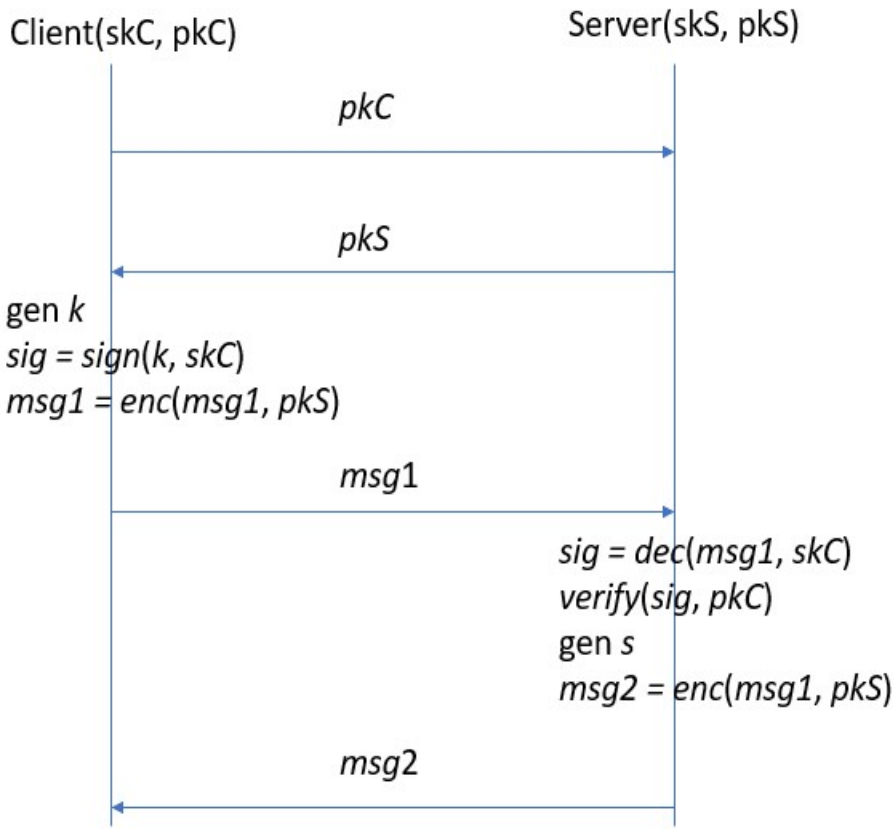
	$<N_A, N_B>_{pk(A)}$ <----------	
- дешифровка сообщения $<N_A, N_B>_{pk(A)}$ - $<N_B>_{pk(I)}$ $<N_B>_{pk(I)}$ ---------->		
	- дешифровка $<N_B>_{pk(I)}$ $<N_B>_{pk(B)}$ ---------->	
(N_A, N_B) -- общий секрет, который известен *I*		

Пример 1: код на Proverif




```
fun Exp(bitstring,bitstring,bitstring):bitstring.
fun Sign(bitstring,bitstring):bitstring.
  reduc forall msg:bitstring,sign_key:bitstring;
    CheckSign(Sign(msg, sign_key), Exp(xCurve, xBase, sign_key)) = msg.
fun Encrypt(bitstring,bitstring):bitstring.
  reduc forall a0:bitstring,a1:bitstring;
    Decrypt(Encrypt(a0,a1),a1) = a0.
fun A_Encrypt(bitstring,bitstring):bitstring.
  reduc forall a0:bitstring,a_key:bitstring;
    A_Decrypt(A_Encrypt(a0,Exp(xCurve, xBase, a_key)),a_key) = a0.
```

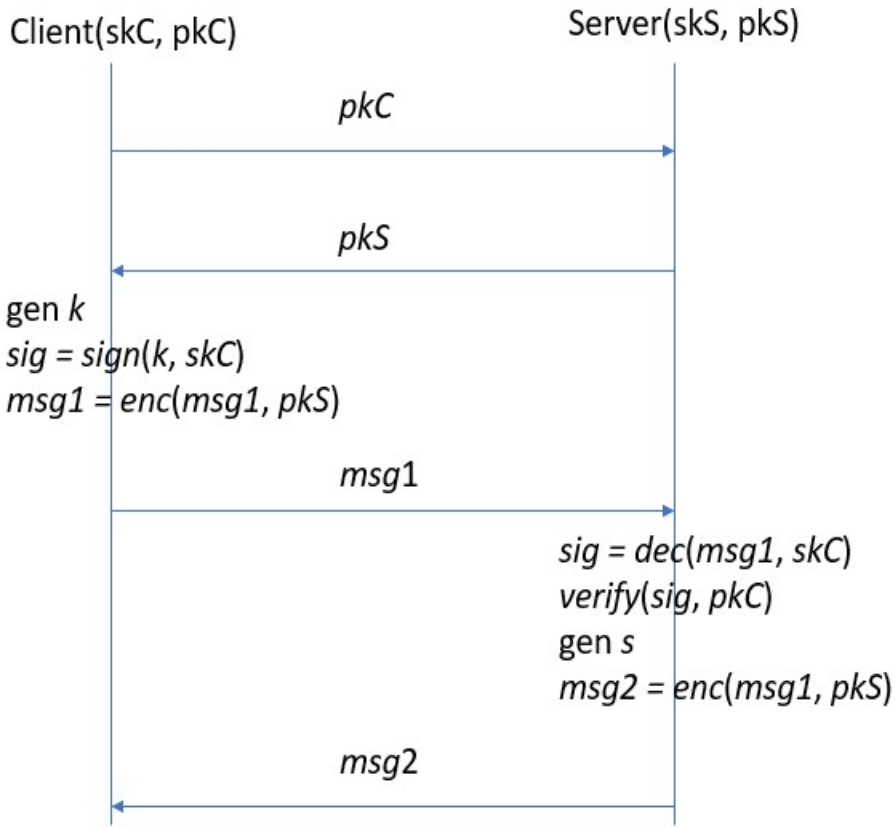
Пример 1: код на Proverif (2)




```
let Client(skC:bitstring, pkC:bitstring) =
  out(c, pkC);
  in(c, pkS:bitstring);
(* msg1 *)
  new k:bitstring;
  let sigC = Sign(k, skC) in
  let msg1 = A_Encrypt(sigC, pkS) in
  out(c, msg1);
(* msg2 *)
  in(c, msg2:bitstring);
  let s = Decrypt(msg2, k) in
  event Client_end(s).
```

```
let Server(skS:bitstring, pkS:bitstring) =
  in(c, pkC:bitstring);
  out(c, pkS);
(* msg1 *)
  in(c, msg1:bitstring);
  let sigC = A_Decrypt(msg1, skS) in
  let k = CheckSign(sigC, pkC) in
(* msg2 *)
  (*new s:bitstring;*)
  let msg2 = Encrypt(s, k) in
  out(c, msg2);
  event Server_end(s).
```

Пример 1: описание протокола дизъюнктами Хорна



Дизъюнкты Хорна, представляющие протокол

$$\text{attacker}(\text{pk}(x)) \Rightarrow \text{attacker}(\text{pencrypt}(\text{sign}(k[\text{pk}(x)], sk_A[]), \text{pk}(x)))$$

$$\text{attacker}(\text{pencrypt}(\text{sign}(y, sk_A[]), \text{pk}(sk_B[]))) \Rightarrow \text{attacker}(\text{sencrypt}(\text{s}, y))$$

Дизъюнкты Хорна, представляющие противника

$$\text{attacker}(m) \wedge \text{attacker}(pk) \Rightarrow \text{attacker}(\text{pencrypt}(m, pk))$$
$$\text{attacker}(sk) \Rightarrow \text{attacker}(\text{pk}(sk))$$
$$\text{attacker}(\text{pencrypt}(m, \text{pk}(sk))) \wedge \text{attacker}(sk) \Rightarrow \text{attacker}(m)$$
$$\text{attacker}(m) \wedge \text{attacker}(sk) \Rightarrow \text{attacker}(\text{sign}(m, sk))$$
$$\text{attacker}(\text{sign}(m, sk)) \Rightarrow \text{attacker}(m)$$
$$\text{attacker}(\text{sign}(m, sk)) \wedge \text{attacker}(\text{pk}(sk)) \Rightarrow \text{attacker}(m)$$
$$\text{attacker}(m) \wedge \text{attacker}(k) \Rightarrow \text{attacker}(\text{sencrypt}(m, k))$$
$$\text{attacker}(\text{sencrypt}(m, k)) \wedge \text{attacker}(k) \Rightarrow \text{attacker}(m)$$
$$\text{attacker}(a[])$$

Выводимо ли событие *attacker*(*s*)?

Пример 1: выводимость события $attacker(s)$

Последовательность применения дизъюнктов Хорна, приводящая к нарушению секретности s

1. $attacker(a[])$
2. $attacker(a[]) \Rightarrow attacker(pk(a[]))$
3. $attacker(pencrypt(sign(k[pk(a[])], skA[]), pk(a[]))) \wedge attacker(a[]) \Rightarrow attacker(sign(k[pk(a[])], skA[]))$
4. $attacker(sign(k[pk(a[])], skA[])) \wedge attacker(pk(skB[])) \Rightarrow attacker(k[pk(a[])])$
5. $attacker(sencrypt(s, k[pk(a[])])) \wedge attacker(k[pk(a[])]) \Rightarrow attacker(s)$

Протокол Диффи-Хеллмана Proverif

- (pr_C, pk_C) – долговременные (закрытый, открытый) ключи клиента
- (x, g^x) – сессионные (закрытый, открытый) ключи клиента
- (pr_S, pk_S) – долговременные (закрытый, открытый) ключи сервера
- (y, g^y) – сессионные (закрытый, открытый) ключи сервера

Диффи-Хеллман

Client (pkC, privC)		**Server (pkS, privS)**
- генерация g^*x*	*<pkC*, g^*x*> ---------->	
	<pkS, g^*y*> <----------	- генерация g^*y*
- *client_key* = g^*y*^*x*		- *server_key* = g^*x*^*y*
key = g^(*x***y*) -- общий секрет		

Противник отправляет серверу сообщение $\overset{<pk_E, g^x>}{\rightarrow}$ от лица клиента

$\Downarrow$

Протокол Диффи-Хэллмана не является стойким по отношении к активному противнику

Протокол Диффи-Хеллмана Proverif

- (pr_C, pk_C) – долговременные (закрытый, открытый) ключи клиента
- (pr_S, pk_S) – долговременные (закрытый, открытый) ключи сервера
- (x, g^x) – сессионные (закрытый, открытый) ключи клиента
- (y, g^y) – сессионные (закрытый, открытый) ключи сервера

Диффи-Хеллман

Client (pkC, privC)		**Server (pkS, privS)**
- генерация g^x	<pkC, g^x> ---------->	
	<pkS, g^y> <----------	- генерация g^y
- client_key = g^y^x		- server_key = g^x^y
key = g^(x*y) -- общий секрет		

Противник отправляет серверу сообщение $\xrightarrow{<pk_E, g^x>}$ от лица клиента

$\Downarrow$

Протокол Диффи-Хэллмана не является стойким по отношении к активному противнику

Tamarin Prover: пример описания протокола BADH

- (pr_C, pk_C) – долговременные (закрытый, открытый) ключи клиента
- (x, g^x) – сессионные (закрытый, открытый) ключи клиента
- $sig_C(mess)$ – ЭЦП на закрытом ключе клиента
- (pr_S, pk_S) – долговременные (закрытый, открытый) ключи сервера
- (y, g^y) – сессионные (закрытый, открытый) ключи сервера
- $sig_S(mess)$ – ЭЦП на закрытом ключе сервера

.

Client (pkC, privC)		**Server (pkS, privS)**
- генерация g^x	<g^x> ---------->	
	<g^y, pkS, sig_S(g^x, g^y)> <----------	- генерация g^y
- client_key = g^y^x	<pkC, sig_C(g^y, g^x)> ---------->	- server_key = g^x^y
key = g^(x*y) -- общий секрет		

Практическое задание

Необходимо построить модель на языке Tamarin протокола **ISO** и прислать её на почту vinevg2015@gmail.com или в телеграмм до 27.11.2020

- (pr_C, pk_C) – долговременные (закрытый, открытый) ключи клиента
- (x, g^x) – сессионные (закрытый, открытый) ключи клиента
- $sig_C(mess)$ – ЭЦП на закрытом ключе клиента
- (pr_S, pk_S) – долговременные (закрытый, открытый) ключи сервера
- (y, g^y) – сессионные (закрытый, открытый) ключи сервера
- $sig_S(mess)$ – ЭЦП на закрытом ключе сервера

Client (pkC, privC)		**Server (pkS, privS)**
- генерация g^x	<*pkC*, g^x> ---------->	
	<*pkS*, g^y, sig_S(g^x, g^y, *pkC*)> <----------	- генерация g^y
- *client_key* = g^y^x	<sig_C(g^y, g^x, *pkS*)> ---------->	- *server_key* = g^x^y
key = g^(x*y) -- общий секрет		