Верификация криптографических протоколов с помощью программы CryptoVerif

Скиренко Андрей

Криптографические протоколы

- Протоколы шифрования
- Протоколы электронной цифровой подписи
- Протоколы аутентификации
- Протоколы аутентифицированного распределения ключей

Верификация криптографических протоколов

- Протоколы шифрования
  - взлом?
- Протоколы электронной цифровой подписи
  - подделываемость?
- Протоколы аутентификации
  - имперсонация?
- Протоколы аутентифицированного распределения ключей
  - перехват?

Разновидности атак

- Атаки, направленные против криптографических алгоритмов
- Атаки против криптографических методов, применяемых для реализации протоколов
- Атаки против самих протоколов (активные или пассивные)

CryptoVerif

- 2002 — создание ProVerif
- 2005 — Первый выпуск CryptoVerif
- 2014 — версия CryptoVerif 1.19

CryptoVerif

- симметричное шифрование
- асимметричное шифрование
- цифровая подпись
- коды подлинности сообщения
- хеш-функции

Использование CryptoVerif

- Full domain hash
- One-Encryption Key Exchange
- Kerberos

Порядок работы

- Описание протокола специализированным языком
- Автоматический анализ с помощью серии игр
- Ручной анализ результатов

Игры

- Применение правил переписывания к протоколу
- Анализ текущего состояния

Структура описания протоколов

- Вспомогательные объявления, криптографические примитивы
- Цели доказательства
- Описание действий участников
- Главное тело процесса

Цели доказательства

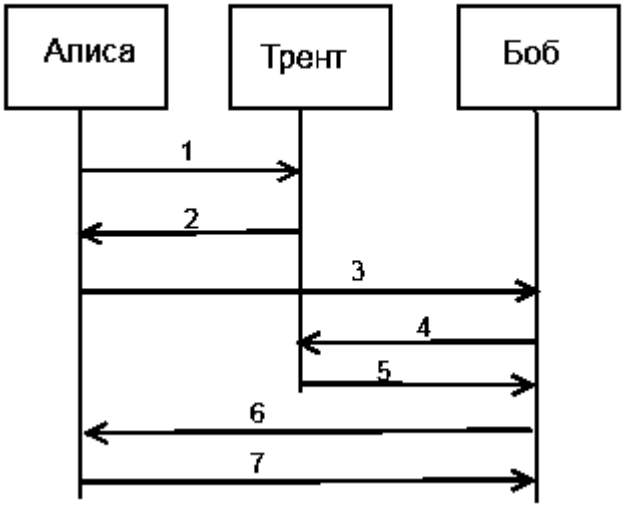
- ```
  query x:host, y:host, na:nonce,
  nb:nonce;

  event endA(x,y,na,nb) ==>
  inj:beginB(x,y,na,nb).
  ```
- ```
  query x:host, y:host, na:nonce,
  nb:nonce;

  event endB(x,y,na,nb) ==>
  inj:beginA(x,y,na,nb).
  ```

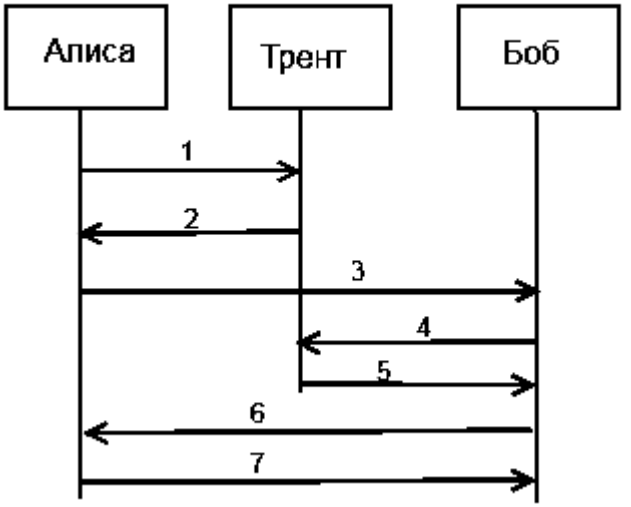
Описание протокола

1. Алиса → A, B → Трент



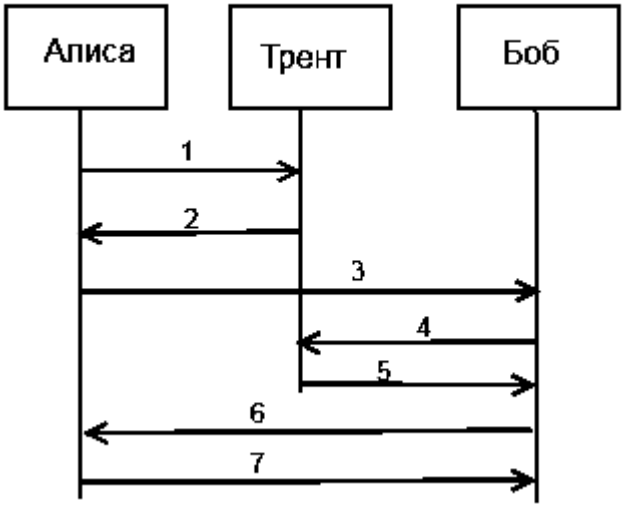
Описание протокола

1. Алиса → A, B → Трент
2. Трент → B, pkB, sT(B, pkB)
   → Алиса



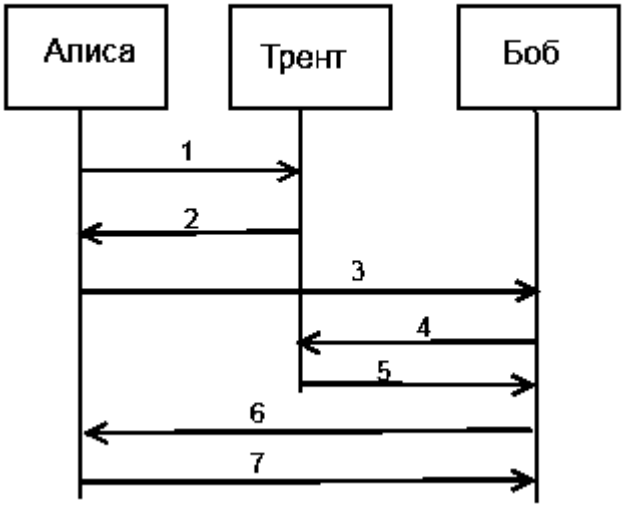
Описание протокола

1. Алиса → A, B → Трент
2. Трент → B, pkB, sT(B, pkB)
   → Алиса
3. Алиса → pkB(Na, A) → Боб



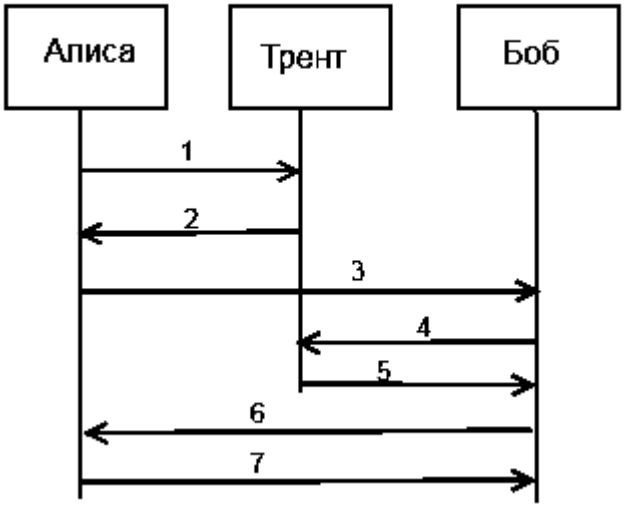
Описание протокола

1. Алиса → A, B → Трент
2. Трент → B, pkB, sT(B, pkB)
   → Алиса
3. Алиса → pkB(Na, A) → Боб
4. Боб → B, A → Трент



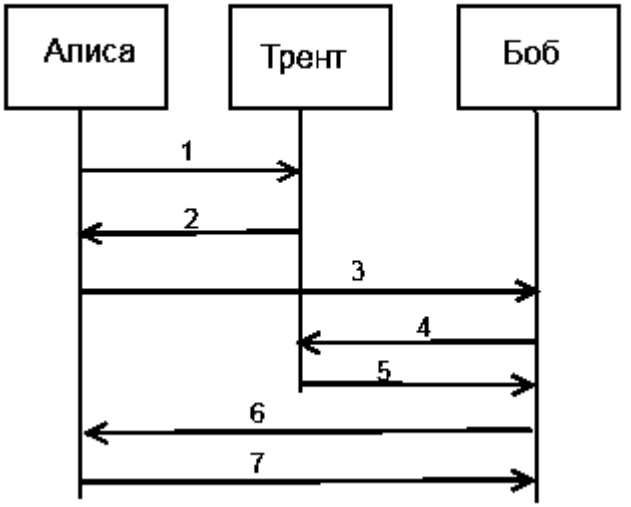
Описание протокола

1. Алиса → A, B → Трент
2. Трент → B, pkB, sT(B, pkB)
   → Алиса
3. Алиса → pkB(Na, A) → Боб
4. Боб → B, A → Трент
5. Трент → A, pkA, sT(A, pkA)



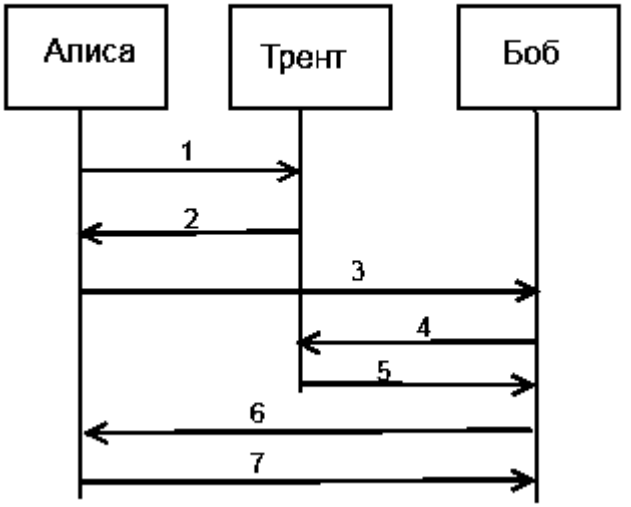
Описание протокола

1. Алиса → A, B → Трент
2. Трент → B, pkB, sT(B, pkB)
   → Алиса
3. Алиса → pkB(Na, A) → Боб
4. Боб → B, A → Трент
5. Трент → A, pkA, sT(A, pkA)
6. Боб → pkA(Na, Nb) → Алиса



Описание протокола

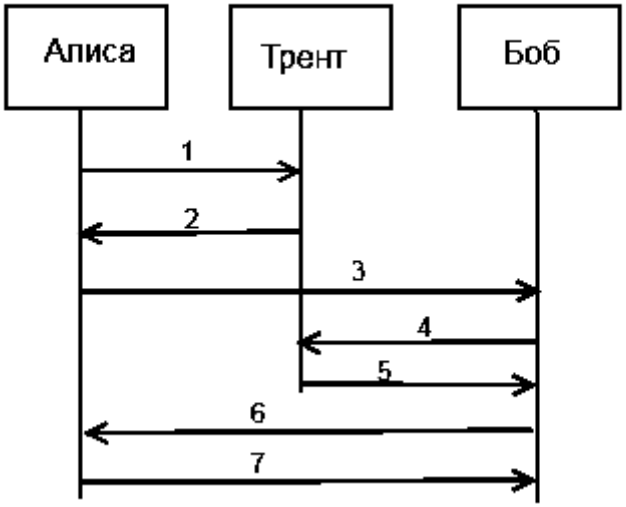
1. Алиса → A, B → Трент
2. Трент → B, pkB, sT(B, pkB)
   → Алиса
3. Алиса → pkB(Na, A) → Боб
4. Боб → B, A → Трент
5. Трент → A, pkA, sT(A, pkA)
6. Боб → pkA(Na, Nb) → Алиса
7. Алиса → pkB(Nb) → Боб



Реализация протокола

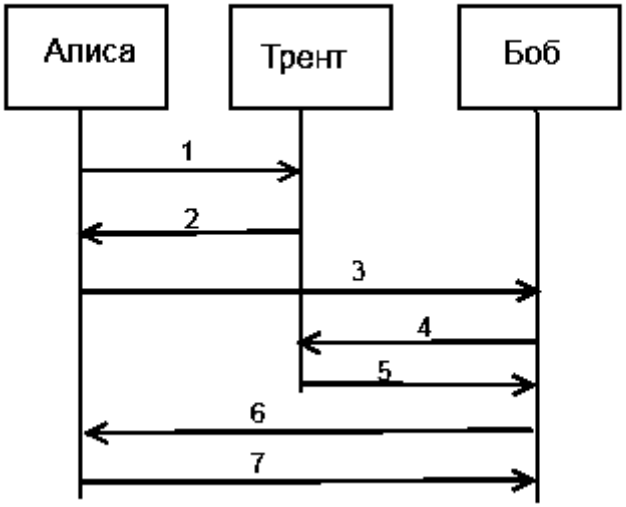
	Алиса	Боб	Трент

1.	`OA1(hostX:host):=`
	
	`return (A, hostX);`
			`OS1(h1: host, h2: host):=`
			
			`find j2 <= N2 suchthat defined(Khost[j2],Rkey[j2]) && (Khost[j2] = h2) then`
			
			`r3 <-R sseed;`



Реализация протокола

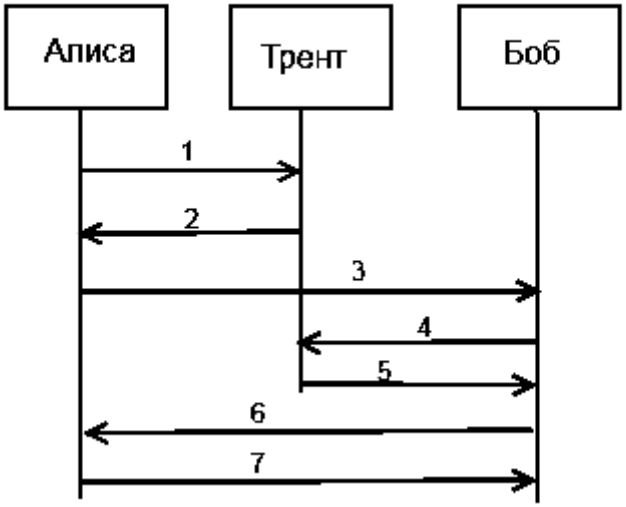
	Алиса	Боб	Трент
2.	OA2(pkX: pkey, =hostX, ms: signature):= if check(concat3(pkX, hostX), pkS, ms) then		return(Rkey[j2], h2, sign(concat3(Rkey[j2], h2), skS, r3)).



Реализация протокола

	Алиса	Боб	Трент

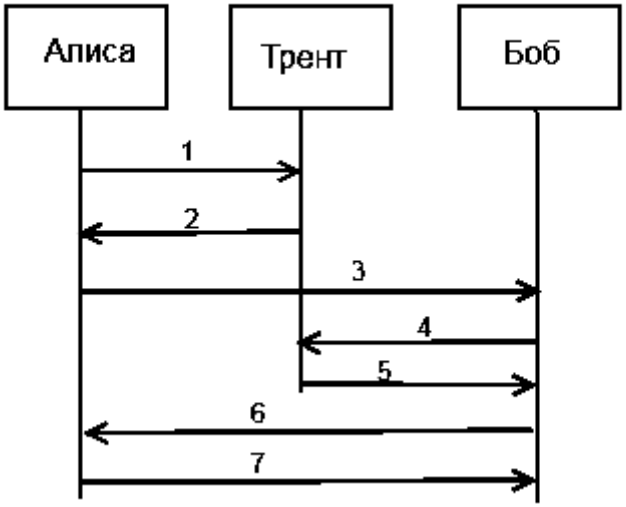
3.	`Na <-R nonce;`
	
	`r3 <-R seed;`
	
	`return(enc(concat1 (Na, A), pkX, r3));`
		`OB1(m:bitstring):=`
		
		`let injbot (concat1(Na, hostY)) = dec(m, skB) in`	



Реализация протокола

	Алиса	Боб	Трент

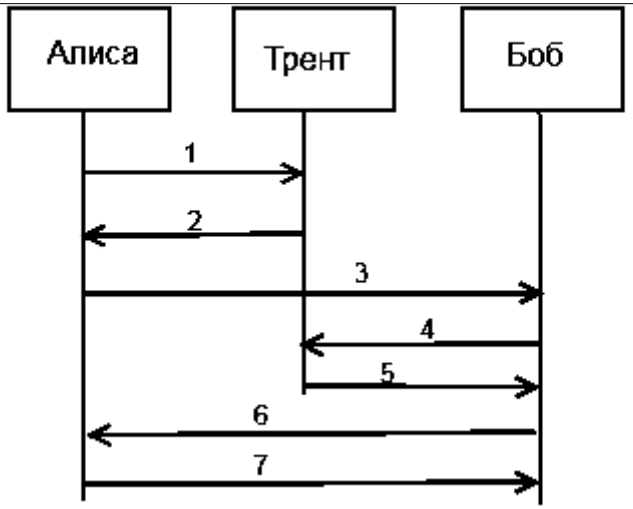
4.		`return(B, hostY);`	`OS1(h1: host, h2: host):=`
			
			`find j2 <= N2`
			`  suchthat`
			`defined(Khost[j2],Rkey[j2]) &&`
			`(Khost[j2] = h2) then`
			
			`r3 <-R sseed;`



Реализация протокола

	Алиса	Боб	Трент

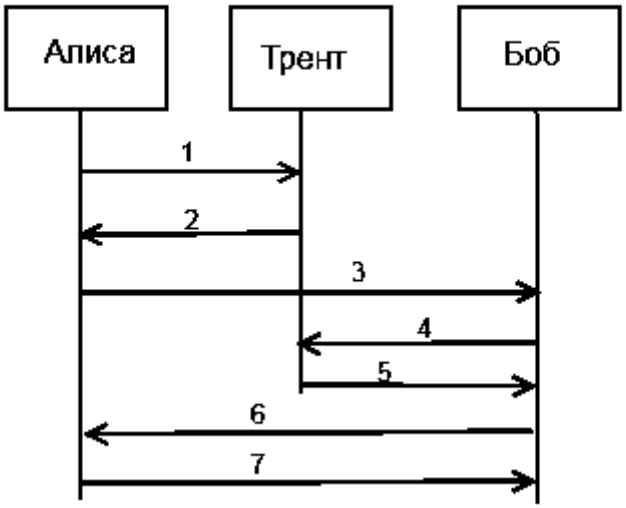
5.		`OB2(pkY: pkey, =hostY, ms:signature):=`
		
		`if check(concat3(pkY, hostY), pkS, ms) then`
			`return(Rkey[j2], h2, sign(concat3(Rkey[j2], h2), skS, r3)).`



Реализация протокола

	Алиса	Боб	Трент

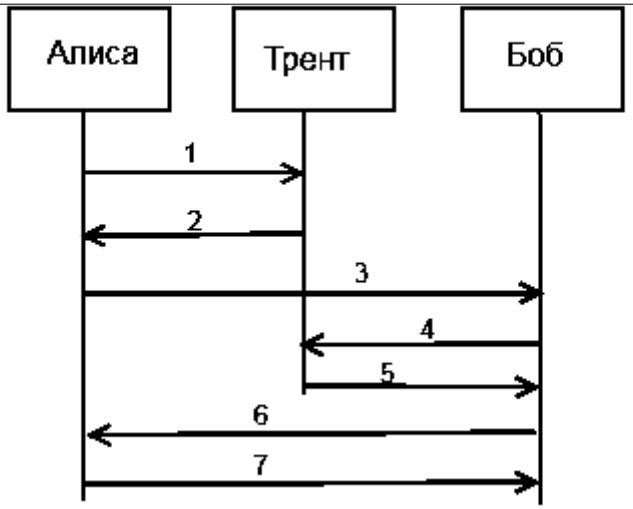
6.	OA3(m: bitstring) :=
	
	let injbot(concat2(=Na, Nb)) = dec(m, skA) in
	
	event beginA(A, hostX, Na, Nb);
		Nb <-R nonce;
		
		event beginB(hostY, B, Na, Nb);
		
		r5 <-R seed;
		
		return(enc(concat4(Na, Nb, B), pkY, r5));	



Реализация протокола

	Алиса	Боб	Трент

7.	`return(enc(pad(Nb), pkX, r4));`
	
	`Ofinish():=`
	
	`if hostX = B then`
	
	`event endA(A, hostX, Na, Nb).`
		`OB3(m3: bitstring):=`
		
		`let injbot(pad(=Nb)) = dec(m3, skB) in`
		
		`if hostY = A then`
		
		`event endB(hostY, B, Na, Nb).`	



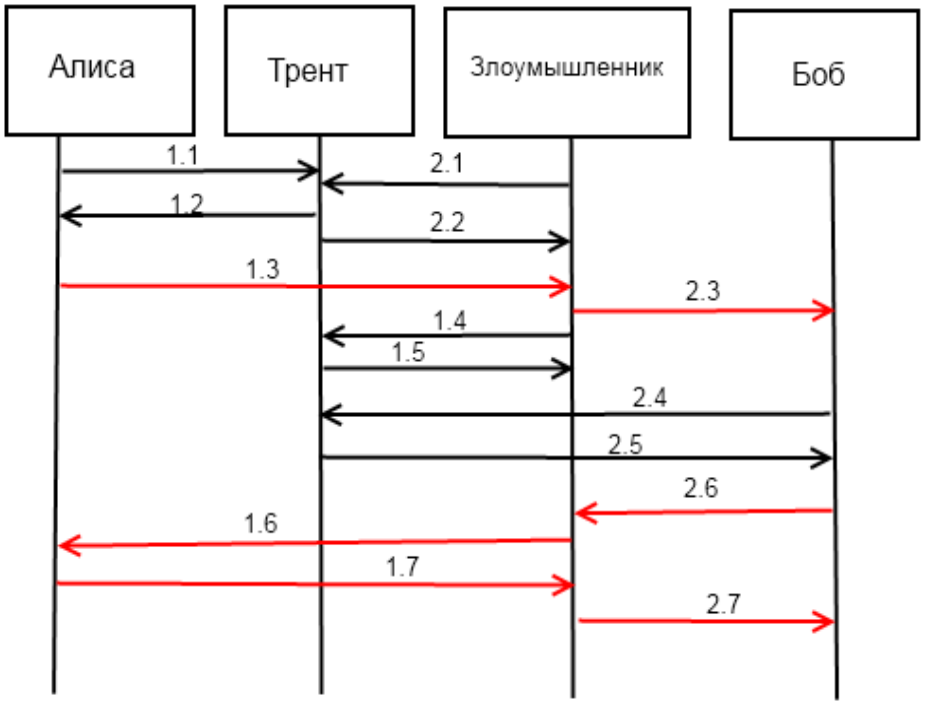
Реализация протокола

```
process
 Ostart():=
 rkA <-R keyseed;
 pkA <- pkgen(rkA); skA <-skgen(rkA);
 rkB <-R keyseed;
 pkB <- pkgen(rkB); skB <- skgen(rkB);
 rkS <-R skeyseed;
 pkS <- spkgen(rkS); skS <- sskgen(rkS);
 return(pkA, pkB, pkS);
 (processA | processB |  (foreach iS <=
N do processS) | (foreach iK <= N2 do
processK))
```

Результат

- RESULT Could not prove event endB(x, y, na, nb) ==> beginA(x, y, na, nb).

Схема атаки на протокол



3. Алиса → pkB(Na, A) → Боб

6. Боб → pkA(Na, Nb) → Алиса

7. Алиса → pkB(Nb) → Боб

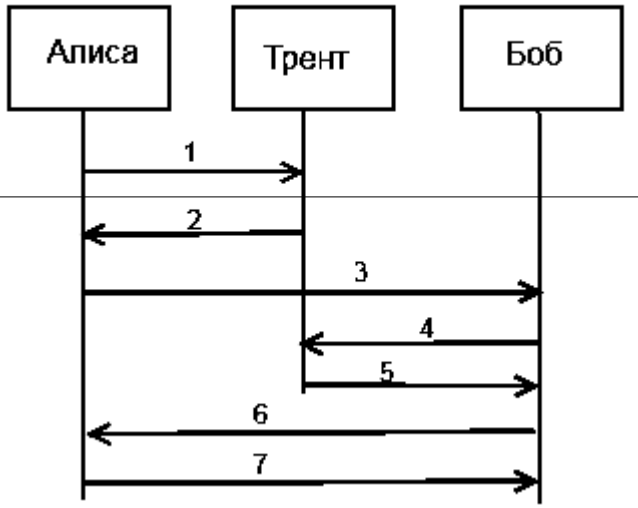
Исправление уязвимости

6. Боб → pkA(Na, Nb, B) → Алиса

Исправление уязвимости

	Алиса	Боб	Трент

6.	OA3(m:bitstring):=
	
	let injbot (concat4(=Na, Nb, =hostX)) = dec(m, skA) in
	
	event beginA(A, hostX, Na, Nb);
		Nb <-R nonce;
		
		event beginB(hostY, B, Na, Nb);
		
		r5 <-R seed;
		
		return(enc(concat4(Na, Nb, B), pkY, r5));	



Результат

```
All queries proved.
```

Библиография

- Bruno Blanchet. A Computationally Sound Automatic Prover for Cryptographic Protocols. In Workshop on the link between formal and computational models, Paris, France, June 2005.
- Bruno Blanchet and David Pointcheval. Automated Security Proofs with Sequences of Games. In Cynthia Dwork, editor, CRYPTO'06, volume 4117 of Lecture Notes in Computer Science, pages 537-554, Santa Barbara, CA, August 2006. Springer.
- Jean-Sébastien Coron. Cryptography, Course no. 10. University of Luxembourg, May 2014.
- Victor Shoup. Sequences of Games: A Tool for Taming Complexity in Security Proofs. Nov. 30, 2004
- Dominique Unruh. Formal Methods and Cryptography. Saarland University. 2010