

**Aria Javani**

**9725303**

**1:**

we have two answers  $x_1, x_2$

$$\begin{cases} x_1 \equiv a \mod m_1 \\ x_1 \equiv b \mod m_2 \end{cases} \text{ and } \begin{cases} x_2 \equiv a \mod m_1 \\ x_2 \equiv b \mod m_2 \end{cases}$$

$$\begin{cases} x_1 \mod m_1 = x_2 \mod m_1 \\ x_1 \mod m_2 = x_2 \mod m_2 \end{cases} \Rightarrow \begin{cases} x_1 - x_2 \equiv m_1 k_1 \\ x_1 - x_2 \equiv m_2 k_2 \end{cases} \Rightarrow k(m_1 m_2) =$$


$$x_1 - x_2 \Rightarrow x_1 \equiv x_2 \mod m_1 m_2$$

**2.1:**

$$p = 467, \alpha = 2, a = 228, b = 57$$

$$k_{prA} = 228, k_{prB} = 57$$

$$k_{pubA} = \alpha^{228} = 394 \mod 467$$

 Modulo calculator

Expression 2^228	Modulus 467	<input type="checkbox"/> Show details
---------------------	----------------	---------------------------------------

CALCULATE

Result  
394

Symmetric representation  
-73

$$k_{pubB} = \alpha^{57} = 313 \mod 467$$

## Modulo calculator

Expression  
 $2^{57}$

Modulus  
467

☐ Show details

CALCULATE

Result  
313

Symmetric representation  
-154

$$k_{AB} = k_{pubA}^{k_{prB}} = k_{pubB}^{k_{prA}} = 2^{228 \times 57} = 2^{12996} = 206 \text{ mod } 467$$

## Modulo calculator

Expression  
 $2^{12996}$

Modulus  
467

☐ Show details

CALCULATE

Result  
206

Symmetric representation  
206

**2.2:**

$$p = 467, \alpha = 4, a = 400, b = 134$$

$$k_{prA} = 400, k_{prB} = 134$$

$$k_{pubA} = \alpha^{400} = 89 \text{ mod } 467$$

## Modulo calculator

Expression  
 $4^{400}$

Modulus  
467


☐ Show details

CALCULATE

Result  
89

Symmetric representation  
89

$$k_{pubB} = \alpha^{134} = 51 \text{ mod } 467$$

 Modulo calculator

Expression  
4^134

Modulus  
467


☐ Show details

CALCULATE

Result  
51

Symmetric representation  
51

$$k_{AB} = k_{pubA}^{k_{prB}} = k_{pubB}^{k_{prA}} = 2^{400 \times 134} = 2^{53600} = 161 \text{ mod } 467$$

 Modulo calculator

Expression  
4^53600

Modulus  
467

☐ Show details

CALCULATE


Result  
161

Symmetric representation  
161

$$p = 467, \alpha = 4, a = 167, b = 134$$

$$k_{prA} = 167, k_{prB} = 134$$

$$k_{pubA} = \alpha^{167} = 89 \text{ mod } 467$$

 Modulo calculator

Expression  
4^167

Modulus  
467


☐ Show details

CALCULATE

Result  
89

Symmetric representation  
89

$$k_{pubB} = \alpha^{134} = 51 \text{ mod } 467$$


 Modulo calculator

Expression 4^134	Modulus 467	<input type="checkbox"/> Show details
---------------------	----------------	---------------------------------------

**CALCULATE**

Result 51	Symmetric representation 51
--------------	--------------------------------

$$k_{AB} = k_{pubA}^{k_{prB}} = k_{pubB}^{k_{prA}} = 2^{167 \times 134} = 2^{22378} = 161 \text{ mod } 467$$

 Modulo calculator

Expression 4^22378	Modulus 467	<input type="checkbox"/> Show details
-----------------------	----------------	---------------------------------------

**CALCULATE**

Result 161	Symmetric representation 161
---------------	---------------------------------

**2.3 :**

Order of our generator is 233 so 167 and 167+233=400 have the same result.

**3 :**

In this attack the attacker tries to send his own public key to Alice and Bob instead of letting them get each other public key. If he does this successfully in the next step Alice or Bob send their messages encrypted with attacker's key so attacker can read them easily.

**4.1 :**

a primitive root of  $p$  is a number  $0 < \alpha < p$  such that all of a powers generate all the number between 0 and  $p$

**5 :**

$$p = 467, g = 2, a = 153 \quad m = 331, k = 197$$

$$k_{pubA} = 2^{153} \bmod 467 = 15 \times 90 \times 8 \bmod 467 = 10800 \bmod 467 = 59$$

$$k_{pubB} = 2^{197} \bmod 467 = 19 \times 90 \times 128 \bmod 467 = 218880 \bmod 467 = 224$$

$$\text{encrypted message: } m_e = k_{pubA}^k m = 59 \times 331 \bmod 467 = 19529 \bmod 467 = 382$$

$$\text{Bob sends } (m_e, k_{pubB}) = (224, 382)$$

$$\text{Alice decryption : } m = \frac{m_e}{k_{pubB}^{153}} = 331$$

**6 :**

we try to show these problems are equivalent and how to convert each one to other.

in elgmal we find message and in diffie hellman we find the key.

$$m = t.r^{-\log_a b} \text{ equation for attacking elgmal}$$

next we need to find the common key in diffie helman

$$k_{AB} = \alpha^{k_{prA}k_{prB}} \text{ by knowing } \alpha^{k_{prA}}, \alpha^{k_{prB}}$$

now suppose we want to know the elgmal equivalence with diffie helman parameters

$$m = 1. (\alpha^{k_{prB}})^{-\log_a \alpha^{k_{prA}}} = \alpha^{-k_{prA}k_{prB}} = k_{AB}^{-1}$$

next we try to find elgmal message by having a algorithm that find diffie hellman key

$$\begin{aligned} \alpha^{k_{prA}k_{prB}} &= \alpha^{k_{prA} \log_a \alpha^{k_{prB}}} \rightarrow \alpha^{k_{prB}} = k_{pubA}, \alpha^{k_{prA}} = k_{AB}, \alpha \\ &= k_{pubB} \end{aligned}$$

by doing this exchanges we can put elgmal parameters in our diffie hellman algorithm.

**7 :**

if we choose 1 then public key is  $\alpha$  which allows attacker to find private key effortlessly.

if we choose  $p-1$  then public key is 1 which means our private key is  $p-1$ .

**8 :**

$$a^p \equiv a$$

we've already proven this in homework 3 q7.2 :

first we assume a set of all possible strings of length  $p$  and a different characters possible for each string so the count of string will be  $a^p$ . among all of these strings there are exactly a strings consisting of exactly one character. with the rest of them we make a necklace with each string

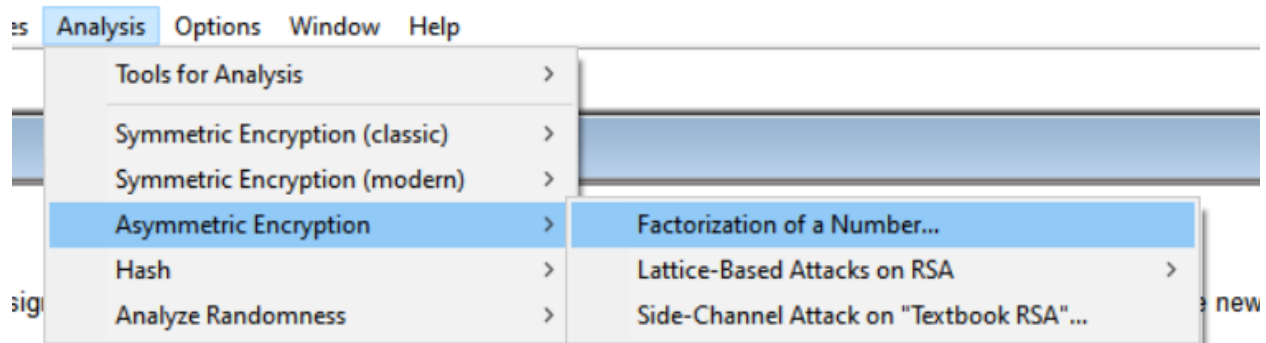
then we consider all the strings with that have a same necklace with subset length of  $p$  in one group the reason we choose  $p$  as size is because a subset of length  $T$  should be chosen with the condition of  $T$  dividing length of whole string and since the length of strings is prim number of  $p$  we shall use  $p$  (also not 1 because it's trivial) as the sub string length. so  $p$  divides  $a^p - a$  and  $a^p = a$

$$(x + y)^p = x^p + y^p$$

$x + y = a$ , using previous theorem :  $a^p \equiv 1, x^p \equiv 1, y^p \equiv 1, 2^p \equiv 1 \rightarrow x^p + y^p = a^p = (x + y)^p$

9 :

$$1) 1963497163 = 33923 \times 57881$$



## Factorization of a Number



Algorithms for factorization

☒ Brute-force  
☒ Brent  
☒ Pollard  
☒ Williams  
☒ Lenstra  
☒ Quadratic sieve

Input

Enter the number to be factorized:

Factorization (stepwise)

Click "Continue" to factor the input number. If the result (shown below) can be factored further, click the button again to execute the factorization.

Factorization

The factorization is represented in the format  $\langle z_1^{a_1} * z_2^{a_2} * \dots * z_n^{a_n} \rangle$ .  
Composite numbers are highlighted in red.

Last factorization through:  Found 2 factors in 0.115 seconds.

Factorization result:  
  
< >

2) prime numbers : 766807766953, 459517077757, 26464987111

Carmichael numbers : 334153, 314821, 294409

regular composite numbers : 111111111, 222222222, 33333333



There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☒ Miller-Rabin test
- ☐ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

766807766953

Result:



766807766953

Test number

Factorize number

Cancel

There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☐ Miller-Rabin test
- ☒ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

766807766953

Result:



766807766953

Test number

Factorize number

Cancel

There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☒ Miller-Rabin test
- ☐ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

459517077757

Result:



459517077757

Test number

Factorize number

Cancel

There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☐ Miller-Rabin test
- ☒ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

459517077757

Result:



459517077757

Test number

Factorize number

Cancel

## Prime Number Test



There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

### Algorithms for prime number test

- ☒ Miller-Rabin test
- ☐ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

### Prime number test

Load number from file

Number or  
formula to test:

26464987111

Result:



26464987111

Test number

Factorize number

Cancel

There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☐ Miller-Rabin test
- ☒ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

26464987111

Result:



26464987111

Test number

Factorize number

Cancel

There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☒ Miller-Rabin test
- ☐ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

334153

Result:



334153

Test number

Factorize number

Cancel

There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☐ Miller-Rabin test
- ☒ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

334153

Result:



334153

Test number

Factorize number

Cancel



There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☒ Miller-Rabin test
- ☐ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

314821

Result:



314821

Test number

Factorize number

Cancel

There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☐ Miller-Rabin test
- ☒ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

314821

Result:



314821

Test number

Factorize number

Cancel

There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☒ Miller-Rabin test
- ☐ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

294409

Result:



294409

Test number

Factorize number

Cancel

There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☐ Miller-Rabin test
- ☒ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

294409

Result:



294409

Test number

Factorize number

Cancel

There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☒ Miller-Rabin test
- ☐ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

111111111

Result:



111111111

Test number

Factorize number

Cancel

## Prime Number Test



There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

### Algorithms for prime number test

- ☒ Miller-Rabin test
- ☐ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

### Prime number test

Load number from file

Number or  
formula to test:

22222222

Result:



22222222

Test number

Factorize number

Cancel

There are many methods to check if a number is prime.

Most of these are probabilistic, meaning that they can only determine primality to a given adjustable degree of certainty.

However, these methods are much faster than their counterpart, deterministic methods. Such methods return a 100% mathematically certain result.

#### Algorithms for prime number test

- ☒ Miller-Rabin test
- ☐ Fermat test
- ☐ Solovay-Strassen test
- ☐ AKS test (deterministic procedure)

#### Prime number test

Load number from file

Number or  
formula to test:

33333333

Result:



33333333

Test number

Factorize number

Cancel

3)

**Algorithm**

☒ **RSA**  
Bit length of RSA modulus:

☐ **DSA**  
Bit length of DSA prime number:


☐ **Elliptic curves**  
Identifier (bit length of curve parameter):

**User data**

The key pair will be put in an encrypted PSE with the name shown below. The key pair will be protected by your PIN code.

Last name:   
First name:   
Key identifier (optional):

**CrypTool**

 The parameters chosen by you and the new key pair have been successfully saved.  
The assigned key identifier is '[Aria][Javani][RSA-1024][1650878568]'.

Elapsed time while creating key pair: 41.210 seconds.

**OK**

The domain parameter of

Parameters	Value of
<input type="text"/>	<input type="text"/>

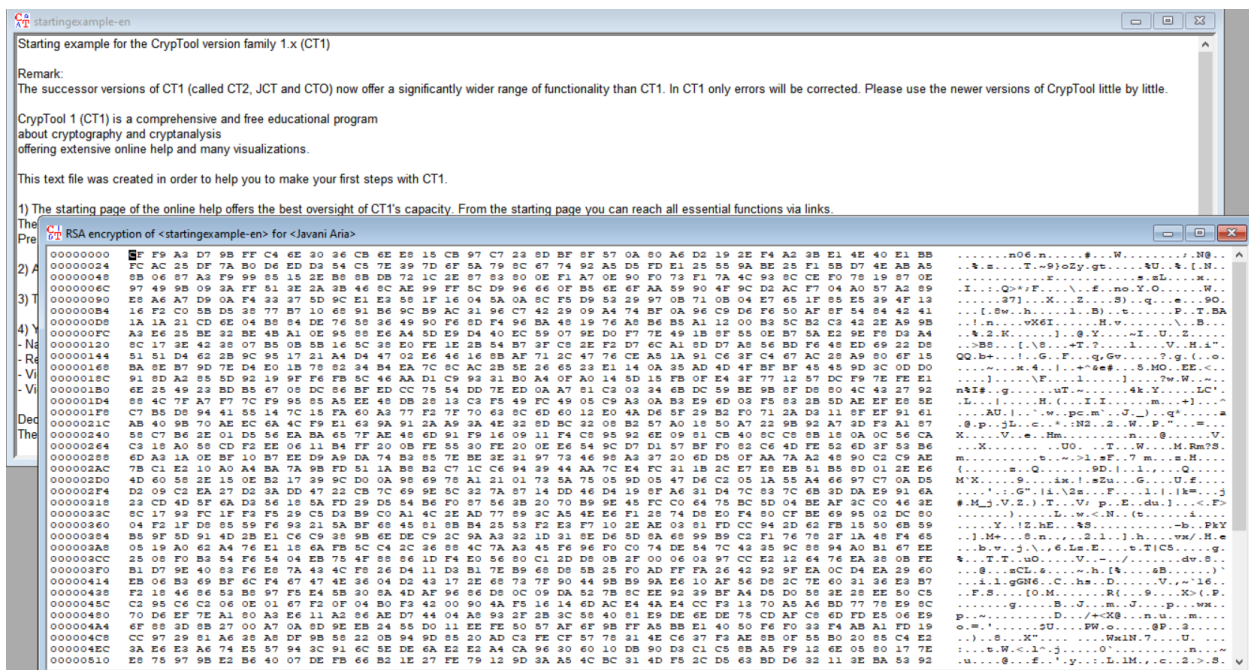
**Base for presentation of numbers**

☐ Octal ☒ Decimal ☐ Hexadecimal

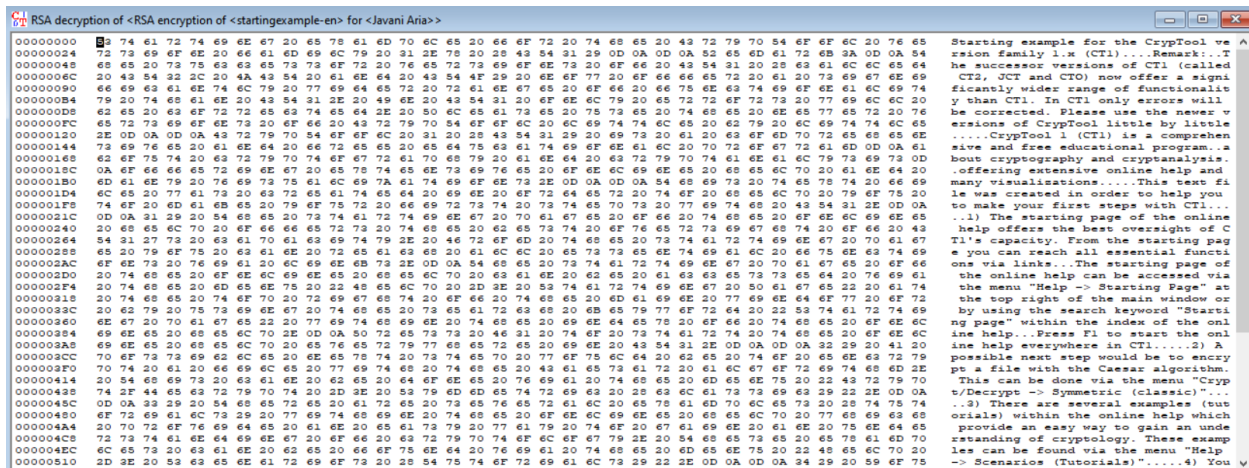
4)

Last name	First name	Key type	Key identifier	Created	Internal ID no.
Aria	Javani	RSA-1024		25.04.2022 02:22:48	1650878568
SideChannelAt...	Bob	RSA-512	PIN=1234	06.07.2006 02:51:34	1152179494





encryption



decryption

5)

Set public parameters

Public parameters:  
Prime module p:   
Generator g:

Set Public Parameters

✕

Both parameters, the generator (g) and the prime module (p) may be known publicly.

The prime module (p) needs to be a prime; if you don't know any large primes, just click the button 'Generate Prime'. A valid prime module will be created instantly.

Prime module:

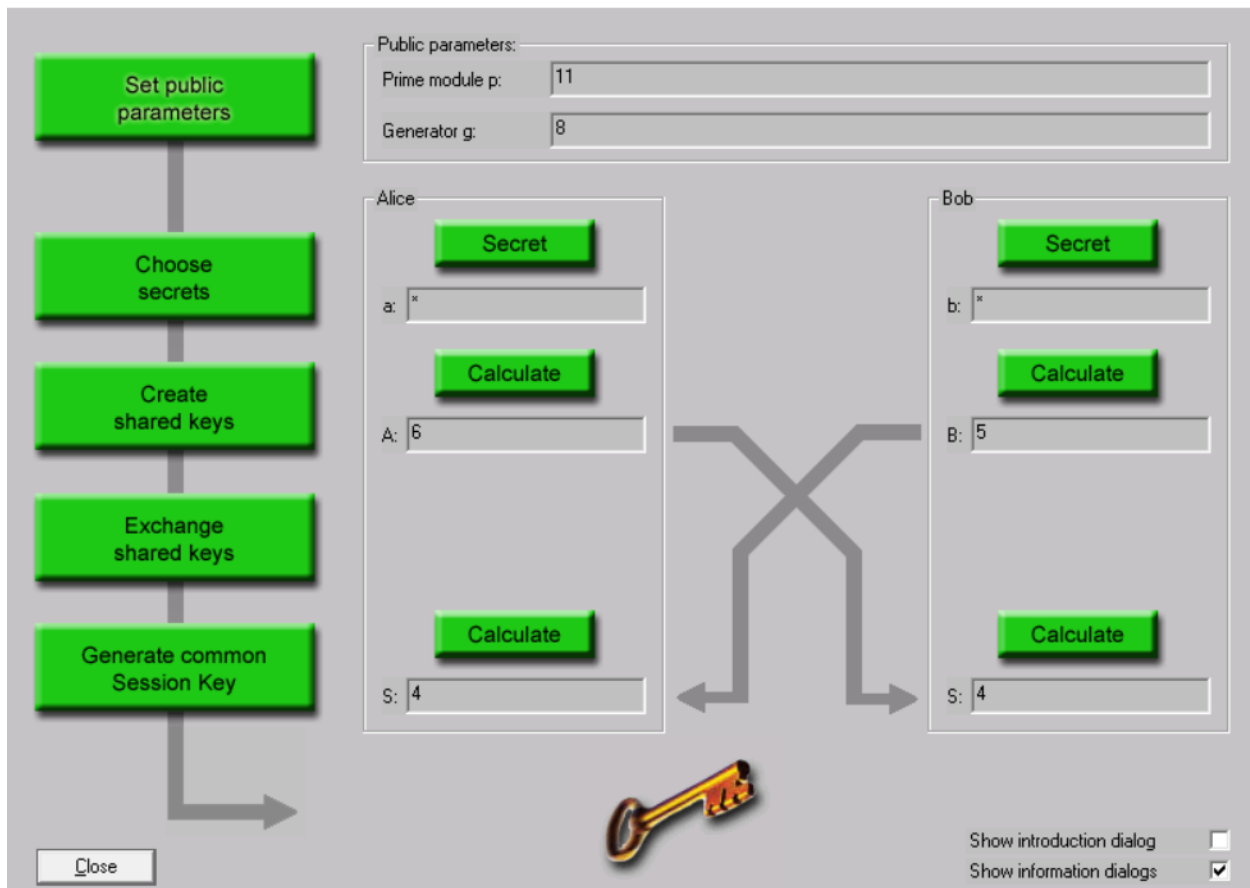
The generator (g) is a natural number, preferably not zero or one and not a multiple of the prime module (p). Push the button 'Create Generator' to make Cryptool create a valid natural number.

Generator:

Close

Show introduction dialog ☐

Show information dialogs ☒



This interface visualizes the Diffie-Hellman Key Exchange Protocol. It features a central workspace with Alice and Bob's interaction panels, a left sidebar with protocol steps, and a top section for public parameters.

**Public parameters:**

- Prime module  $p$ : 11
- Generator  $g$ : 8

**Protocol Steps (Left Sidebar):**

- Set public parameters
- Choose secrets
- Create shared keys
- Exchange shared keys
- Generate common Session Key

**Alice's Panel:**

- Secret:** (Green button)
- a:** (Input field with 'x')
- Calculate:** (Green button)
- A:** 6 (Output field)
- Calculate:** (Green button)
- S:** 4 (Output field)

**Bob's Panel:**

- Secret:** (Green button)
- b:** (Input field with 'x')
- Calculate:** (Green button)
- B:** 5 (Output field)
- Calculate:** (Green button)
- S:** 4 (Output field)

**Key Exchange:** Arrows show Alice's 'A' (6) being sent to Bob and Bob's 'B' (5) being sent to Alice.

**Session Key:** Both Alice and Bob have calculated a common session key  $S = 4$ , represented by a key icon.

**Controls:**

- Close:** (Button)
- Show introduction dialog:** ☐
- Show information dialogs:** ☒