Substitution Ciphers and Block Ciphers

- A **cipher** takes a **plaintext** and **encodes** it to produce a **ciphertext**.
- In many cryptographic systems, the letters of the alphabet are represented by the numbers 0, 1, ..., 25.
- In a **substitution cipher**, characters in the plaintext are replaced by other characters.
- **Affine transformations** can be used to produce simple substitution ciphers.
- In a **block cipher**, blocks of characters are replaced by blocks of characters.
- **Affine matrix transformations** can be used to produce simple block ciphers.

A **cipher** takes a message (the **plaintext**) and **encodes** it — puts it in a form (the **ciphertext**) where the information in the message is not obvious upon inspection. The recipient of the message takes the ciphertext and **decodes** it — performs an operation which recovers the plaintext from the ciphertext.

Example. (A **shift cipher**) This is also known as a **Caesar cipher**, since it was supposedly used by Julius Caesar.

The letters of the alphabet will be represented by the numbers 0, ..., 25:

$$A = 0, B = 1, C = 2, \ldots, Z = 25.$$

I won't make a distinction between upper and lower case. If x is a letter, I'll encode it using

$$y = x + 11 \pmod{26}.$$

(I could use any nonzero number from 1 to 25 in place of 11.) The formula above replaces each letter with another letter; in effect, the alphabet gets "shifted" 11 places to the left.

The translation table for this cipher is:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K

For example, consider the message

I WAS ASLEEP AT THE TIME

"I" is replaced by "T", "W" by "H", and so on. I get the ciphertext

T HLD LWDPPA LE ESP ETXP

I'll group the letters into 5-letter words to hide the original word groupings:

THLDL DWPPA LEESP ETXPZ

I've thrown in an extra "Z" at the end to make the last group come out evenly. If you decode the message, you'd get

IWASA SLEEP ATTHE TIMEO

You can see that the 5-letters grouping and the extra "Z" did no harm, since it's evident what the plaintext was.

Note that the equation $y = x + 11 \pmod{26}$ can be solved for x:

$$x = y + 15 \pmod{26}.$$

This equation is the decoding transformation; it's equivalent to reading the translation table backward. □

Example. If you use the shift

$$y = x + 19 \pmod{26},$$

the message

I MUST HAVE FOOD becomes B FNLM ATOX YHHW.

I wrote a computer program to do this. There are only 26 possible shifts, so if you wanted to decode this by brute force, you could feed the ciphertext through 26 shift programs and see which one produced a sensible message. Shift ciphers are not of much use when it comes to protecting secrets! □

The next thing to try is an **affine transformation**:

$$y = ax + b \pmod{26}, \text{ where } (a, 26) = 1.$$

I need the last condition in order to ensure that I can decode messages. This is equivalent to being able to invert the transformation. Now if $(a, 26) = 1$, then a is invertible mod 26, so

$$x = a^{-1}(y - b) \pmod{26}.$$

This equation can be used to decode messages.

Example. Consider the transformation $y = 5x + 14 \pmod{26}$. To find the decoding transformation, solve for x in terms of y:

$$y = 5x+14 \pmod{26}, \quad y+12 = 5x \pmod{26}, \quad 21(y+12) = 21 \cdot 5x \pmod{26}, \quad x = 21y+18 \pmod{26}.$$

Here's the translation table:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
O	T	Y	D	I	N	S	X	C	H	M	R	W	B	G	L	Q	V	A	F	K	P	U	Z	E	J

The plaintext

WHEN WILL I BECOME A FISH

gives the ciphertext

UXIB UCRR C TIYGWI O NCAX

Group the letters:

UXIBU CRRCT IYGWI ONCAX □

Shift ciphers and affine transformation ciphers are called **substitution** or **character ciphers** because each letter is replaced by another letter. They're simple to use, but relatively easy to crack. For example, with any reasonably large message you can count the letters in the ciphertext and guess the substitution using frequency tables for letters in the English language.

As a partial remedy to frequency analysis, you might think of enciphering blocks of k letters at a time. To do this, encode letters as number from 0 to 25 in the usual way. Consider a block of k letters $a_1 a_2 \dots a_k$. As the cipher key, choose a $k \times k$ matrix M which is invertible mod 26. (M will be invertible mod 26 if $\det M$is relatively prime to 26.) Then the cipher transformation is $\vec{c} = M\vec{a} \pmod{26}$, i.e.

$$\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \end{bmatrix} = M \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_k \end{bmatrix} \pmod{26}.$$

You can decipher messages using $\vec{a} = M^{-1}\vec{c} \pmod{26}$.

Example. (a **digraphic cipher**) Take

$$M = \begin{bmatrix} 5 & 3 \\ 2 & 3 \end{bmatrix}.$$

Since $\det M = 9$ and $(9, 26) = 1$, M is invertible mod 26. In fact,

$$M^{-1} = \frac{1}{9}\begin{bmatrix} 3 & -3 \\ -2 & 5 \end{bmatrix} = 3 \cdot \begin{bmatrix} 3 & 23 \\ 24 & 5 \end{bmatrix}.$$

(Note that $\dfrac{1}{9} = 9^{-1} = 3$, because $3 \cdot 9 = 1 \pmod{26}$.)
Here's a message:

SPICY MEATBALLS

Break it up into two-letter groups and convert them to 2-dimensional vectors:

SP	IC	YM	EA	TB	AL	LS
$(18, 15)$	$(8, 2)$	$(24, 12)$	$(4, 0)$	$(19, 1)$	$(0, 11)$	$(11, 18)$

Finally, use M to encode each vector. For example,

$$\begin{bmatrix} 5 & 3 \\ 2 & 3 \end{bmatrix}\begin{bmatrix} 18 \\ 15 \end{bmatrix} = \begin{bmatrix} 5 \\ 3 \end{bmatrix} \pmod{26}.$$

$(5, 3) =$ FD, so the first two letters of the ciphertext are "FD".
Continuing in this way, you obtain the ciphertext

FD UW AG UI UP HH FY □

Exponentiation Ciphers and Public Key Cryptography

- An **exponentiation cipher** encodes messages using $C = A^e \pmod{p}$, where A is a block of the plaintext and C is the ciphertext. They are less vulnerable to frequency analysis than character ciphers or simple block ciphers.
- In a **public-key system**, the **public key** is known to everyone and can be used to encipher messages. The **private key** is only known by an individual, and can be used to decipher messages.
- The **RSA cryptosystem** is a public-key system which uses an exponentiation cipher. It depends for its security on the difficulty of factoring large numbers.

Exponentiation ciphers are due to Pohlig and Hellman (1978). They are less vulnerable to frequency analysis than block ciphers. Here's the procedure.

1. Let p be a prime number, and let e be the **exponent**. They should satisfy $(e, p-1) = 1$.
2. Encode the letters of the alphabet as

A	B	$\cdots$	Z
00	01	$\cdots$	25

3. Group the letters in the message in blocks of m letters, where m is chosen so that

$$\underbrace{2525\cdots25}_{m \text{ times}} < p < \underbrace{2525\cdots25}_{(m+1) \text{ times}}$$

   For example, suppose $p = 4283$. Then you should use blocks of $m = 2$ letters, because $2525 < 4283 < 252525$. And if $p = 670417$, you should use blocks of $m = 3$ letters, because $252525 < 670417 < 25252525$. This stipulation merely ensures that the blocks are unique mod p.

4. Encode a block A using

$$C = A^e \pmod{p}.$$

The ciphertext C is an integer satisfying $0 \le C < p$ and this integer *is* the ciphertext: You don't convert it to letters.

Example. Let $p = 2621$, so $p - 1 = 2620 = 2^2 \cdot 5 \cdot 131$. Take an exponent relatively prime to 2620 — for instance, $e = 11$.

I use blocks of $m = 2$ letters, because

$$2525 < 2621 < 252525.$$

Take the plaintext and convert it to numbers:

DE	EP	YO	GU	RT
0304	0415	2414	0620	1719

Now encode the message:

$$(0304)^{11} = 0065 \pmod{2621}$$

$$(0415)^{11} = 0415 \pmod{2621}$$

$$(2414)^{11} = 1323 \pmod{2621}$$

$$(0620)^{11} = 1567 \pmod{2621}$$

$$(1719)^{11} = 0150 \pmod{2621}$$

The ciphertext is

$$00650415132315670150$$

How should you do these computations? The *best* way to do the computations is to use a computer mathematics program, such as *Mathematica* or *Derive*. Most calculators can only accomodate 10–20 digit integers. If you try to compute 2414^{11} on a calculator, you'll find that it's around 1.62×10^{37}. Because these computations require modular arithmetic, you can't use floating point — you are losing significant digits.

So how do you do something like 2414^{11} if all you have is a calculator? First, rewrite it:

$$2414^{11} = (2414^2)^5 \cdot 2414.$$

Now I'll compute 2414^2 and reduce it mod 2621:

$$2414^2 = 5827396, \quad 5827396 = 913 \pmod{2621}.$$

(I got the last result by finding $\dfrac{5827396}{2621} \approx 2223.34834$. Subtract the integer part (2223) times 2621 from 5827396: $5827396 - 2223 \cdot 2621 = 913$.)

Therefore,

$$2414^{11} = (2414^2)^5 \cdot 2414 = 913^5 \cdot 2414 \pmod{2621}.$$

It should be clear how to proceed. Use the rules for exponents to reduce the product a little bit at a time, so that the intermediate results don't overflow your calculator.

As I said, it is easier to use a computer! □

To decode a message that has been encoded using an exponentation cipher, find d such that

$$de = 1 \pmod{p-1}.$$

This is possible (using the Euclidean algorithm), since $(e, p-1) = 1$ by assumption. Equivalently, $de = 1 + k(p-1)$ for some k. Now suppose $C = A^e \pmod{p}$. Then

$$C^d = A^{de} = A^{1+k(p-1)} = A \cdot A^{k(p-1)} = A \cdot (A^{p-1})^k = A \cdot 1^k = A \pmod{p}.$$

Note that A is less than $2525 \cdots 25$ (m 25's) because A came from a block of m letters. Since $2525 \cdots 25 < p$, it follows that $p \nmid A$, and little Fermat applies. Thus, $A^{p-1} = 1 \pmod{p}$.

In other words, raising C to the d-th power recovers the plaintext from the ciphertext.

Example. Take $p = 2621$ and $e = 11$. $(2620, 11) = 1$; apply the Extended Euclidean algorithm:

a	q	y
2620	-	1191
11	238	5
2	5	1
1	2	0

$$1 = (-5) \cdot 2620 + 1191 \cdot 11.$$

Hence, $1191 \cdot 11 = 1 \pmod{2620}$.
So to decode $C = 1407$, raise it to the 1191-th power:

$$(1407)^{1191} = 0712 \pmod{2621}.$$

$0712 = \text{HM}$, which is the plaintext for this block. □

In a **public-key cryptosystem**, there are separate keys for encoding and decoding messages. One key is public, so that anyone can send a message to me. But I'm the only one who knows the private key, so I'm the only one who can read my messages. Moreover, I can use my private key to *send* messages, which can be decoded using the public key. Since I'm the only one who could have encoded such a message, people know the message must have come from me — a **digital signature**.

I'll discuss the **RSA** public-key cryptosystem, which is due to Rivest, Shamir, and Adleman (1978). You'll see that it's essentially a modified exponentiation cipher.

1. Let p and q be *large* prime numbers. For practical applications, you'll need primes which are around 100 digits long. Let $n = pq$. (n is called the **key**.)

2. Find an exponent e such that $(e, \phi(n)) = 1$, and such that $2^e > n$.

   If n were prime, $\phi(n)$ would be $n - 1$, and I'd have the setup for an exponentiation cipher. The condition $2^e > n$ guarantees that you can't recover the plaintext A by taking e-th roots. For if A is any block besides $0 \cdots 00$ or $0 \cdots 01$, the result is $> n$ when it's raised to the e-th power, so it changes when it's reduced mod n.

3. Encode the letters of the alphabet as

A	B	$\cdots$	Z
00	01	$\cdots$	25

4. Group the letters in the message in blocks of m letters, where m is chosen so that

   $$\underbrace{2525 \cdots 25}_{m \text{ times}} < n < \underbrace{2525 \cdots 25}_{(m+1) \text{ times}}$$

5. Encode a block A using

   $$C = A^e \pmod{n}.$$

Example. Let $n = 37 \cdot 71 = 2627$. Then

$$\phi(n) = \phi(37)\phi(71) = 36 \cdot 70 = 2520.$$

I can choose e to be any number relatively prime to 2520, and such that $2^e > 2627$. I'll take $e = 13$.
Since $2525 < 2627 < 252525$, I use blocks of two letters.
Take the plaintext and convert it to numbers:

CR	AB	LE	GS
0217	0001	1104	0618

Now encode the message:

$$(0217)^{13} = 1652 \pmod{2627}$$

$$(0001)^{13} = 0001 \pmod{2627}$$

$$(1104)^{13} = 1400 \pmod{2627}$$

$$(0618)^{13} = 1839 \pmod{2627}$$

The ciphertext is

$$1652000114001839 \quad \square$$

When this system is used, e and n are made public so people can encipher messages. The security of this method depends on the difficulty of finding $\phi(n)$, since (as I'll show below) this is what you need to decode a message.

On the one hand, if you know p and q, then

$$\phi(n) = \phi(p)\phi(q) = (p-1)(q-1).$$

Since p and q are known, so is $\phi(n)$.

On the other hand, suppose you know $\phi(n)$, you *don't* know p and q, but you *do* know that n is a product of two primes p and q. Then

$$\phi(n) = \phi(p)\phi(q) = (p-1)(q-1) = pq - p - q + 1 = n - p - q + 1.$$

Therefore,

$$p + q = n - \phi(n) + 1.$$

Moreover,

$$p - q = \sqrt{(p+q)^2 - 4pq} = \sqrt{(p+q)^2 - 4n}.$$

The last two equations show that if you know $\phi(n)$ (and n), then you can find $p+q$, and from that you can find $p-q$. But

$$p = \left(\frac{1}{2}(p+q) + \frac{1}{2}(p-q)\right) \quad \text{and} \quad q = \left(\frac{1}{2}(p+q) - \frac{1}{2}(p-q)\right).$$

Thus, you know p and q.

To summarize, knowing $\phi(n)$ is equivalent to knowing p and q.

If p and q are 100-digit primes, then $n = pq$ is around 200 digits. With present technology, it's hard to factor an arbitrary 200-digit number. It follows that finding $\phi(n)$ — and hence, breaking the code — is difficult at the moment, which means the system is fairly secure.

Of course, *no* cipher is immune to human carelessness! If you let someone discover your key, the cipher is worthless.

Now here's how knowing $\phi(n)$ allows you to decode a message. The idea is similar to that used in the exponentiation cipher.

Find d such that

$$de = 1 \pmod{\phi(n)}.$$

This is possible (using the Euclidean algorithm), since $(e, \phi(n)) = 1$ by assumption. Equivalently, $de = 1 + k\phi(n)$ for some k. Now suppose $C = A^e \pmod{n}$. Then

$$C^d = A^{de} = A^{1+k\phi(n)} = A \cdot A^{k\phi(n)} = A \cdot (A^{\phi(n)})^k = A \cdot 1^k = A \pmod{n}.$$

$A^{\phi(n)} = 1$ is a consequence of Euler's theorem, and will be true provided $(A, n) = 1$. Now $n = pq$, so it's possible for this to fail if the plaintext A has either p or q as a prime factor. However, if p and q are each around 100 digits long, the probability that this will happen is around 10^{-99} — so it's nothing to worry about.

Just as in the exponentiation cipher, raising C to the d-th power recovers the plaintext from the ciphertext.

Example. Take $n = 2627$ and $e = 13$. I'll show that 2114 is *not* an enciphered message by decoding it. Recall that $\phi(2627) = 2520$. Apply the Extended Euclidean algorithm:

a	q	y
2520	-	1163
13	193	6
11	1	5
2	5	1
1	2	0

$$(6)(2520) + (-1163)(13) = 1, \quad \text{so} \quad (-1163)(13) = 1 \pmod{2520}, \quad \text{and} \quad 1357 \cdot 13 = 1 \pmod{2520}.$$

Then

$$(2114)^{1357} = 1980 \pmod{2627}.$$

However, 80 can't be a block in a message, because it's greater than 25. Therefore, 2114 is not a ciphertext for this key. □

Attacking RSA

Nikita's public key is (n, e). If we compute the factorization of $n = pq$, then we can compute $\varphi(n)$ and hence deduce her secret decoding number d. Thus attempting to factor n is a way to try to break an RSA public-key cryptosystem. In this lecture we consider several approaches to "cracking" RSA, and relate them to the difficulty of factoring n.

1 Factoring n Given $\varphi(n)$

If you know $\varphi(n)$ then it is easy to factor n:
Suppose $n = pq$. Given $\varphi(n)$, it is very easy to compute p and q. We have

$$\varphi(n) = (p-1)(q-1) = pq - (p+q) + 1,$$

so we know both $pq = n$ and $p + q = n + 1 - \varphi(n)$. Thus we know the polynomial

$$x^2 - (p+q)x + pq = (x-p)(x-q)$$

whose roots are p and q. These roots can be found using the quadratic formula.

Example 1.1.

```
? n=nextprime(random(10^10))*nextprime(random(10^10));
? phin=eulerphi(n);
? f = x^2 - (n+1-phin)*x + n
%6 = x^2 - 12422732288*x + 31615577110997599711
? polroots(f)
%7 = [3572144239, 8850588049]
? n
%8 = 31615577110997599711
? 3572144239*8850588049
%9 = 31615577110997599711
```

2 When p and q Are Close

Suppose that p and q are "close" to each other. Then it is easy to factor n using a factorization method of Fermat.

Suppose $n = pq$ with $p > q$, say. Then

$$n = \left(\frac{p+q}{2}\right)^2 - \left(\frac{p-q}{2}\right)^2.$$

Since p and q are "close",

$$s = \frac{p-q}{2}$$

is small,

$$t = \frac{p+q}{2}$$

is only slightly larger than $\sqrt{n}$, and $t^2 - n = s^2$ is a perfect square. So we just try

$$t = \text{ceil}(\sqrt{n}), \quad t = \text{ceil}(\sqrt{n}) + 1, \quad t = \text{ceil}(\sqrt{n}) + 2, \ldots$$

until $t^2 - n$ is a perfect square s^2. Then

$$p = t + s, \qquad q = t - s.$$

Example 2.1. Suppose $n = 23360947609$. Then

$$\sqrt{n} = 152842.88\ldots.$$

If $t = 152843$, then $\sqrt{t^2 - n} = 187.18\ldots$.
If $t = 152844$, then $\sqrt{t^2 - n} = 583.71\ldots$.
If $t = 152845$, then $\sqrt{t^2 - n} = 804 \in \mathbb{Z}$.
Thus $s = 804$. We find that $p = t + s = 153649$ and $q = t - s = 152041$.

Here is a bigger example in PARI:

```
? q=nextprime(random(10^50))
%20 = 781770964442308045040751227924107493547437128808803
? p=nextprime(q+1)  \\ a nearby prime
%21 = 781770964442308045040751227924107493547437128808899
? n=p*q
%22 = 611165840845056469708563420184597685050990858094998688952570 4...
      ...259650342157399279163289651693722481897
? t=floor(sqrt(n))+1
  ***   precision loss in truncation
? \p150        \\ set precision of floating-point computations.
   realprecision = 154 significant digits (150 digits displayed)
? t=floor(sqrt(n))+1
%29 = 781770964442308045040751227924107493547437128808851
? sqrt(t^2-n)
%30 = 48.000000000000000000000000000000000000000000000000000000000....
? s=48
%31 = 48
? t + s     \\ p
%33 = 781770964442308045040751227924107493547437128808899
? t - s     \\ q
%35 = 781770964442308045040751227924107493547437128808803
```

1 Public-key Cryptography


Nikita must communicate vital information to Michael, who is a thousand kilometers away. Their communications are being monitored by The Collective, which must not discover the message. If Nikita and Michael could somehow agree on a secret encoding key, they could encrypt their message. Fortunately, Nikita knows about an algorithm developed by Diffie and Hellman in 1976.

2 The Diffie-Hellman Key Exchange Protocol

Nikita and Michael agree on a prime number p and an integer g that has order $p-1$ modulo p. (So $g^{p-1} \equiv 1 \pmod{p}$, but $g^n \not\equiv 1 \pmod{p}$ for any positive $n < p-1$.) Nikita chooses a random number $n < p$, and Michael chooses a random number $m < p$. Nikita sends $g^n \pmod{p}$ to Michael, and Michael sends $g^m \pmod{p}$ to Nikita. Nikita can now compute the secret key:

$$s = g^{mn} = (g^m)^n \pmod{p}.$$

Likewise, Michael computes the secret key:

$$s = g^{mn} = (g^n)^m \pmod{p}.$$

Now Nikita uses the secret key s to send Michael an encrypted version of her critical message. Michael, who also knows s, is able to decode the message.

Meanwhile, hackers in The Collective see both $g^n \pmod p$ and $g^m \pmod p$, but they aren't able to use this information to deduce either m, n, or $g^{mn} \pmod p$ quickly enough to stop Michael from thwarting their plans. Yeah!

The Diffie-Hellman key exchange is the first public-key cryptosystem every published (1976). The system was discovered by GCHQ (British intelligence) a few years before Diffie and Hellman found it, but they couldn't tell anyone about their work; perhaps it was discovered by others before. That this system was discovered independently more than once shouldn't surprise you, given how simple it is!

2.1 Some Quotes

A review of Diffie and Hellman's groundbreaking article is amusing, because the reviewer, J.S. Joel, says "They propose a couple of techniques for implementing the system, but the reviewer was unconvinced."

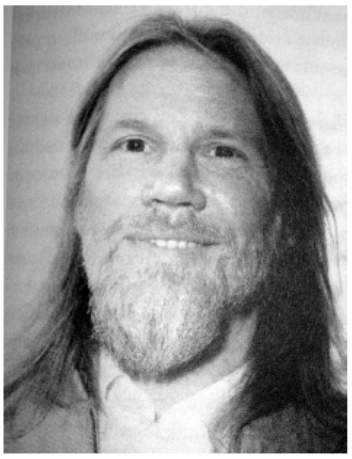
```
Diffie, Whitfield; Hellman, Martin E.
 New directions in cryptography.
 IEEE Trans. Information Theory IT-22 (1976), no. 6, 644--654.
```

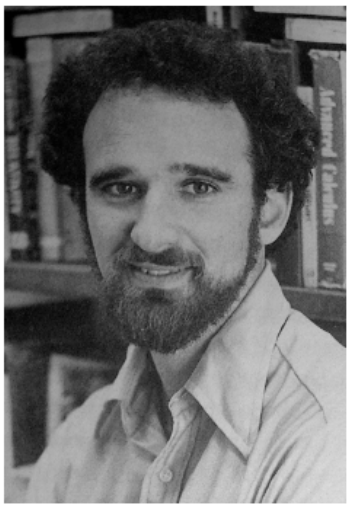
> The authors discuss some of the recent results in communications theory that have arisen out of the need for security in the key distribution channels. They concentrate on the use of ciphers to restrict the extraction of information from a communication over an insecure [channel]. As is well known, the transmission and distribution is then likely to become a problem, in efficiency if not in security. The authors suggest various possible approaches to avoid these further problems that arise. The first they call a "public key distribution system", which has the feature that an unauthorized "eavesdropper" will find it computationally infeasible to decipher the message since the enciphering and deciphering are governed by distinct keys. They propose a couple of techniques for implementing the system, but the reviewer was unconvinced.

Somebody named Alan Westrope wrote in 1998 about political implications:

> The 1976 publication of "New Directions in Cryptography", by Whitfield Diffie and Martin Hellman, was epochal in cryptographic history. Many regard it as the beginning of public-key cryptography, analogous to a first shot in what has become an ongoing battle over privacy, civil liberties, and the meaning of sovereignty in cyberspace.

Here is what Diffie and Hellman look like, respectively:





3 Let's try it!

To make finding g easier, let's choose a prime p such that $(p-1)/2 = q$ is prime (so $p - 1 = 2q$, with q prime). Since for any g with $\gcd(g, p) = 1$,

$$g^{2q} \equiv 1 \pmod{p},$$

the order of g is 1, 2, q, or $2q = p - 1$, so the order of g is easy to compute.

For our first example, let $p = 23$. Then $g = 5$ has order $p - 1 = 22$. (I found $g = 5$ using the function `znprimroot` in PARI. You can also just compute the order of 2, 3, etc., until you find a number with order $p - 1$.)

Nikita: Chooses secret $n = 12$; sends $g^{12} = 5^{12} \equiv \mathbf{18} \pmod{23}$.

Michael: Chooses secret $n = 5$; sends $g^5 = 5^5 \equiv \mathbf{20} \pmod{23}$.

Compute Shared Secret:
Nikita: $20^{12} \equiv \mathbf{3} \pmod{23}$
Michael: $18^5 \equiv \mathbf{3} \pmod{23}$.