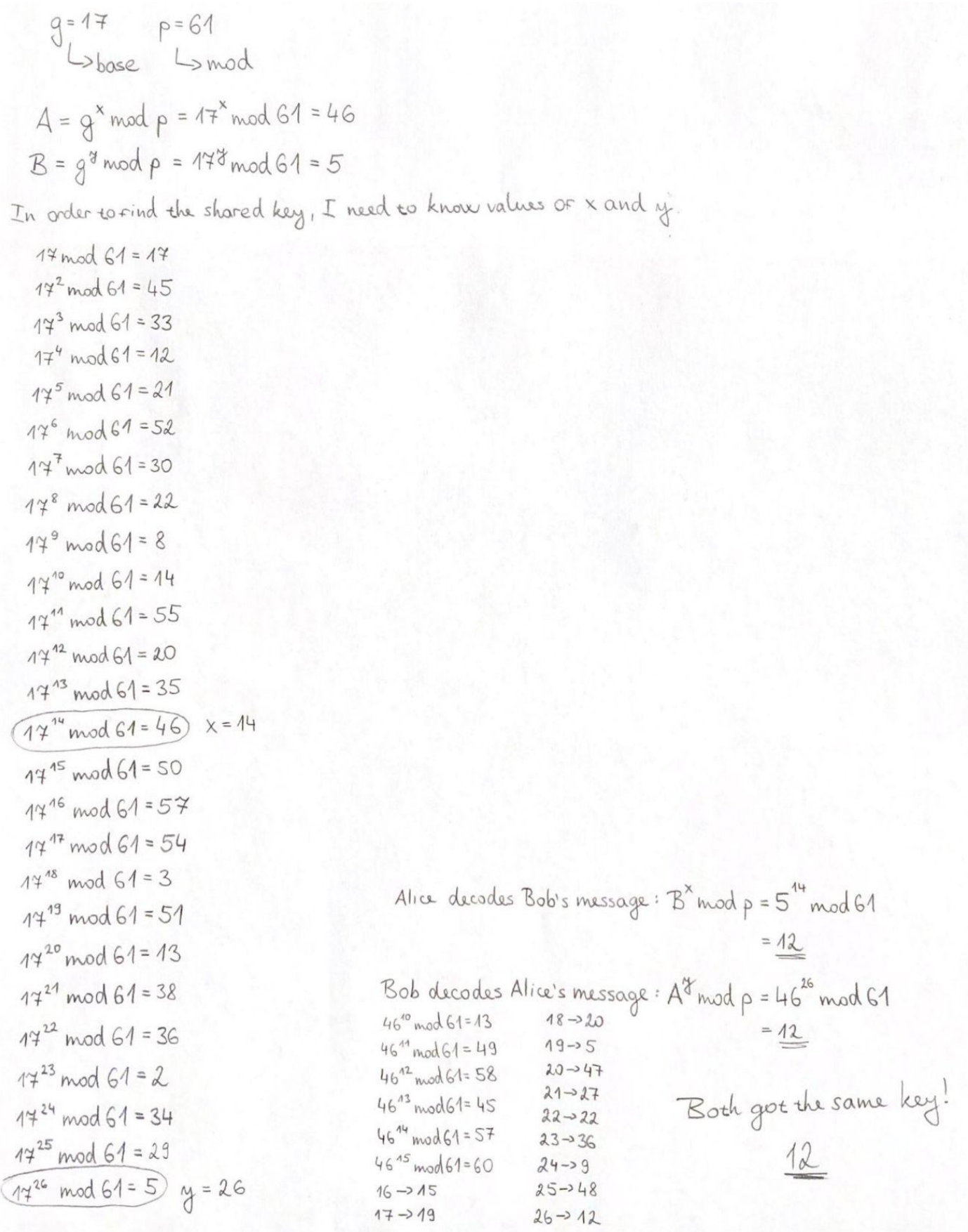
Adela Dujsikova & Ethan Li
CS 231

Being Eve

In the picture below, you can see our scratch work for solving the Diffie Hellman problem.



Adela Dujsikova & Ethan Li
CS 231

In this problem, the values of p, g, A, and B were given. The first step was to express A and B in terms of p and q, so that we get two equations, one with only x as the unknown and the other with only y as the unknown. Since there is no efficient way to solve for x and y, we used the brute force method and started substituting integers from 1 to 60 for x and y (both x and y have to be smaller than p, which is 61). For each of these integers, we calculated (17^x mod 61) and (17^y mod 61) and compared the results to the actual A and B exchanged between Alice and Bob, until we found the correct values of x and y. It turned out that Alice's secret number, x, is 14, and Bob's secret number, y, is 26.

Once we know the secret numbers, we can proceed to decode the "shared secret" agreed upon by Alice and Bob. Alice decodes Bob's message by raising it to the power of her secret number and finding its mod p. This yields the answer of 12. Bob can use the same technique to decode Alice's message, he raises it to the power of his secret number and finds its mod p. His decoding also yields the answer 12. This means that the secret numbers we found were correct, because using them for decoding the message gave us the same result on both sides.

Note: When solving this problem, we haven't yet realized that Python is able to compute the high powers and its modulo, so we devised a method to use a regular calculator to do this. It relies on knowing the modulo of the previous power, that's why you see the one by one calculations when Bob is decoding Alice's message in the scratch work. The method we used is as follows:

1. Calculate the biggest possible power the calculator is able to do, in this case it was $46^{10} mod\ 61 = 1$.
2. Then you can do $(13 \cdot 46)\ mod\ 61$which is the same thing as calculating $46^{11} mod\ 61$. Both of these equations are equal to 49.
3. To find the result of $46^{12} mod\ 61$you just have to follow the same pattern. Take the result of the previous calculation, 49, and do $(49 \cdot 46)\ mod\ 61$to get 58.

Using Python for the calculations is much more effective, but we think this is a nice method to use if one only has a calculator at hand.

Answers to the questions

1. The secret exchanged between Alice and Bob is 12.
2. Our work is shown above.
3. In the brute force step of figuring out both x and y. Instead of a list of 26 calculations, it would be much longer and take significantly more time. We suppose that all Diffie Hellman exchanges are technically solvable, but from the practical point of view it could take years if not decades to find the right x and y. That's why having big enough values of x and y (and p) would make the conversation very secure. The current practice of the key exchange in RSA typically uses 1024 or 2048 bits long integers, which are about 300 or 600 digits long prime numbers, which are much, much, much larger than our 2-digit prime numbers 17 and 61.

Adela Dujsikova & Ethan Li
CS 231

RSA

For the second problem, we are given that Bob's public key is $(e_B, n_B) = (31, 4661)$. In order to decode the message from Alice, the only thing we need to know is d_B, which is a part of Bob's private key.

First, we need to realize that n_B is a product of p_B and q_B, which are two prime numbers. Since n_B ends in 1, p_B and q_B must end either both in 1 or both in 9. We looked at the list of prime numbers and after a few computations, we were able to conclude that p_B and q_B have to be 59 and 79, respectively (although it doesn't really matter which one is which). Next, we looked at the equation Bob uses to determine the value of d_B, which is $e_B d_B mod (p_B - 1)(q_B - 1) = 1$. We can substitute e_B, p_B, and q_B into the equation to get
$31 d_B mod\ (58 \cdot 78) = 31 d_B mod\ 4524 = 1$.

Now we needed to evaluate d_B by a combination of brute force and finding patterns. If d_B is smaller than 4524, $31 d_B mod\ 4524$would simply be equal to whatever $31 d_B$is, which cannot be equal to 1 for any integer d_B. Therefore, the first value we were interested in was the one that would make $31 d_B$bigger than 4524. A simple calculation, $4524 \div 31$, shows that 146 is the smallest d_B to satisfy this. By using 146 in the equation:

$$31 \cdot 146\ mod\ 4524 = 2$$

This result means that we have to keep looking for the next $31 d_B$ values, since the answer, 2, is already bigger than 1, and increasing d_B until $31 d_B$is bigger than $2 \cdot 4524$ would only make the remainder even larger. Therefore, we have to move on to the $31 d_B$ values that are larger than $2 \cdot 4524$, and the smallest value to satisfy that is $d_B = 292$. By trying 292 in the equation:

$$31 \cdot 292\ mod\ 4524 = 4$$

Again, the value of the remainder is already larger than 1 and not what we are looking for. Following the same reasoning, the next value for d_B we tried was such the smallest d_B value that would make $31 d_B$bigger than $3 \cdot 4524$, which is 438:

$$31 \cdot 438\ mod\ 4524 = 6$$

We noticed that each time we "surpassed" another 4524, the result increased by 2. This means that the smallest possible remainder of $31 d_B mod\ 4524$bigger than $4 \cdot 4524$would be 8, and if $31 d_B$is be bigger than $5 \cdot 4524$, the smallest possible remainder would be 10, and so on.

Adela Dujsikova & Ethan Li
CS 231

You can notice the pattern that the smallest possible remainder is always twice the number by which 4524 is multiplied.

Because d_b is always multiplied by 31, the smallest remainders will go in cycles. That's why to get to the result of 1, we have to follow this pattern until the results get over 31. Specifically, in this case the results increase by two with the growing number of multiplications of 4524. This means that the result which comes right before 1 is 30. We know that the result is twice the number by which we multiply 4524, therefore to get 30 we need to multiply 4524 by 15, and to get 1 we need to multiply 4524 by 16. Now we can apply the previously described knowledge, which means $d_B = (16 \cdot 4524 + 1)/31 = 2335$. We can also check this value by plugging it back into the equation:

$$31 \cdot 2335 \, mod \, 4524 = 1$$

This way we discovered that the value of d_B in Bob's private key is 2335. Knowing this information, we wrote a simple Python program and applied the decoding formula $y^{d_B} mod \, n_B$ on each of the numbers in Alice's encoded message and printed the decoded message.

```
message = [2677, 4254, 1152, 4645, 4227, 1583, 2252, 426, 3492, 4227, 3889
    , 1789, 4254, 1704, 1301, 4227, 1420, 1789, 1821, 1466, 4227, 2252,
    3303, 1420, 2234, 4227, 4227, 1789, 1420, 1420, 4402, 1466, 4070, 3278
    , 3278, 414, 414, 414, 2234, 1466, 1704, 1789, 2955, 4254, 1821, 4254,
    4645, 2234, 1704, 2252, 3282, 3278, 426, 2991, 2252, 1604, 3278, 1152,
    4645, 1704, 1789, 1821, 4484, 4254, 1466, 3278, 1512, 3602, 1221, 1872
    , 3278, 1221, 1512, 3278, 4254, 1435, 3282, 1152, 1821, 2991, 1945,
    1420, 4645, 1152, 1704, 1301, 1821, 2955, 1604, 1945, 1221, 2234, 1789
    , 1420, 3282, 2991, 4227, 4410, 1821, 1301, 4254, 1466, 3454, 4227,
    4410, 2252, 3303, 4645, 4227, 3815, 4645, 1821, 4254, 2955, 2566, 3492
    , 4227, 3563, 2991, 1821, 1704, 4254]
toPrint = ""
for num in message:
    res = (num**2335)%4661
    toPrint += chr(res)
print(toPrint)
```

```
Dear Bob, Check this out.  https://www.schneier.com/blog/archives/2017/12/e
    -mail_tracking_1.html Yikes! Your friend, Alice
>
```

Answers to the questions

1. The message Alice sent to Bob was "Dear Bob, Check this out. https://www.schneier.com/blog/archives/2017/12/e-mail_tracking_1.html Yikes! Your friend, Alice".
2. Our work is shown above.
3. Finding p and q. If n_B was much larger (say, 500 digits long prime numbers instead of the two 2-digit ones we used), it would be very hard and time consuming to find the two prime numbers that n_B is the product of.