Exercise Sheet 0x04

PSI-IntroSP: Introduction to Security and Privacy

Privacy and Security in Information Systems Group
https://www.uni-bamberg.de/informatik/psi/

Please submit your solutions using VC (https://vc.uni-bamberg.de) as one PDF file. We strongly encourage you to work in teams on the task sheets. The recommended team size is two students. The maximum acceptable team size is three students. Please include the names and student numbers of all group members in your solutions. It is sufficient that one team member submits the sheet.

Late submissions will be accepted according to the following policy. If you submit up to 12 hours late, you can only achieve 50 % of the maximum points obtainable. For submissions that are late more than 12 hours you can only achieve 25 % of the maximum points. Submissions that arrive later than 24 hours after the deadline will not score any points at all.

Feel free to ask for help on Mattermost while solving the following tasks! If your Mattermost account does not work, try the "reset password" function[1] first. If that does not help, send an email to Henning <henning.pridoehl@uni-bamberg.de>.

Task 1: Number Theory and Algebra — 20 Points

In the first task we will look into basic number theory and algebra. This includes primes, (cyclic) groups, rings, finite fields, inverses of their elements, Fermat's little theorem and more.

A prime is a number greater than 1 that is only divisible by itself and by 1. There are an infinite number of them. Prime numbers are involved in many cryptography operations. They allow us to form cyclic groups and finite fields, which will be explained shortly. Any integer n can be uniquely represented by a product of primes. This representation is said to be the *factorization* of n. To factor an integer is a hard (computational expensive) problem and the basis for many cryptographic protocols.

Calculations are performed in algebraic structures. Those structures define operations on elements and their combination. The most important algebraic structures are semigroups, monoids, (albelian) groups (for one operation), and rings and fields (two operations).

A semigroup is a set S together with a binary operation "$\cdot$". The operation is associative, i. e., $(a \cdot b) \cdot c = a \cdot (b \cdot c)$, where a, b and c are elements of S. The operation forms a closure, i. e. if a and b are elements of S, then is $a \cdot b$. A monoid is a semigroup, which additionally has a neutral element e, i. e., $a \cdot e = e \cdot a = a$. A group is a monoid for which all elements have inverses a^{-1}, i. e., $a \cdot a^{-1} = a^{-1} \cdot a = e$. An abelian group is a group, which is commutative, i. e., where $a \cdot b = b \cdot a$. These definitions might sound strange first, but they are useful. You have already seen monoids and (abelian) groups,

[1] https://mattermost.psi.h4q.it/reset_password

and handled them naturally when calculating with integers and rationals. Multiplication on integers $(\mathbb{Z}, \cdot)$ is a monoid. The neutral element is 1 (no surprise) and there are no inverses except 1, which is the inverse of 1. You can not find a number ($\neq 1$), that if multiplied with another number (in the integers) results in the neutral element 1. If we go to the rationals (without 0), we get a group. All elements have inverses: the reciprocal, i. e., the inverse of a is $\frac{1}{a}$. We will see that we can also have multiplication on integers with inverses, if we slightly modify the operation.

Regarding algebraic structures with two operations, we will start with a ring. A ring is a set R with two operations, "$+$" and "$\cdot$". $(R, +)$ is an abelian group, while $(R, \cdot)$ is a monoid. Multiplication ("$\cdot$") is distributive, i. e., $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$. In a field $(R, +)$ is an abelian group and $(R \setminus \{0\}, \cdot)$ is an abelian group. Every field is a ring, therefore it is distributive too. RSA uses a ring called $\mathbb{Z}_n$ where n is a product of two large primes p and q. Some other cryptographic schemes use finite fields over integers, which are denoted by $\mathbb{F}_p$.

Explain why the integers $\mathbb{Z}$ form a ring, and the rationals $\mathbb{Q}$ form a field.

We will now look into a group, the multiplication of integers modulo n, denoted $\mathbb{Z}_n^*$. It contains the elements $\{1, \ldots, n-1\}$ for which the greatest common divisor (gcd) between the element and n is 1. In this group, integers are multiplied as usual, but for the final result you divide the result by n and take the remainder. Example: Assume $n = 8$ and you want to multiply 5 and 7. This results in 35. If you divide by 8, you get $4 \cdot 8 + 3 = 35$, so the remainder is 3, which is our final result. In other words: $5 \cdot 7 = 3$ in $\mathbb{Z}_8^*$. The condition on the elements, namely for an element a it must hold $gcd(a, n) = 1$, makes sure, that the element has an inverse. Trivially, 1 has always an inverse, which is 1. The inverse of 5 in $\mathbb{Z}_8^*$ is 5, since $5 \cdot 5 = 1$ in $\mathbb{Z}_8^*$. While 5 has an inverse in $\mathbb{Z}_8^*$, 4 does not, since $gcd(4, 8) = 2 \neq 1$ and therefore it is not an element in $\mathbb{Z}_8^*$.

Find the inverses of all elements in $\mathbb{Z}_7^*$. Why do all numbers between 1 and 6 have an inverse?

If the integer n is a prime p all numbers between 1 and $p - 1$ have inverses. To find the inverse by calculation rather than trying, we will look into a algorithm called the *extended euclidean algorithm*. There exists $x, y \in \mathbb{Z}$ for which $a \cdot x + b \cdot y = \gcd(a, b)$ (called Bézout's identity). If $\gcd(a, b) = 1$, we can see that x is the inverse of a modulo b, since modulo b is expressed by "$+ y \cdot b$". The extended euclidean algorithm calculates x and y and therefore the inverse. How it works can be best seen by looking at an example: We want the inverse of 7 in $\mathbb{Z}_{40}^*$. We start by calculating the greatest common divisor of 7 and 40. Split them into divisor and remainder recursively, and keep the results:

$$
\begin{aligned}
40 &= 5 \cdot 7 + 5 \\
7 &= 1 \cdot 5 + 2 \\
5 &= 2 \cdot 2 + 1 \\
2 &= 2 \cdot 1 + 0
\end{aligned}
$$

We repeat this, until the remainder is 0. In this case, the divisor and therefore greatest common divisor is 1, which means, that there exists an inverse.

Now we start to plug in our results in reverse and expand in between to get it into the form $a \cdot x + b \cdot y = 1$, starting with the equation before the last equation:

$$
\begin{aligned}
1 &= 5 - 2 \cdot 2 \\
1 &= 5 - 2 \cdot (7 - 1 \cdot 5) \\
1 &= 3 \cdot 5 - 2 \cdot 7 \\
1 &= 3 \cdot (40 - 5 \cdot 7) - 2 \cdot 7 \\
1 &= 3 \cdot 40 - 17 \cdot 7
\end{aligned}
$$

The inverse of 7 in $\mathbb{Z}_{40}^*$ is -17. This sounds strange, since -17 is not in the set of the monoid. We have not mentioned yet, that for *cyclic groups*, which we will see in a moment, there are multiple elements representing the same element. To get the canonical form (i. e., the one we usually use), add 40 until you are in the range 1 to 39. In this case, -17 + 40 = 23. So 23 is the inverse of 7 in $\mathbb{Z}_{40}^*$ (in canonical form)!

As an exercise, calculate the inverse of 331 in $\mathbb{Z}_{1234}^*$ using the extended euclidean algorithm.

A group G is cyclic if it can be generated by a single element g, noted $G = \langle g \rangle$. The element g is called the *generator*. Each element can be written as a power of g. For example, 3 is a generator of $\mathbb{Z}_7^*$, since $G = \langle 3 \rangle = \{3^1, 3^2, 3^3, 3^4, 3^5, 3^6\} = \{3, 2, 6, 4, 5, 1\}$. Another generator of $\mathbb{Z}_7^*$ is 5. The order of elements when generated by a generator with increasing powers is arbitrary, which is a property used in cryptography. There is no easy way to find the generators of a group. For small groups, you can guess a generator and try all possible powers[2] up to the group order (number of elements in the group, in $\mathbb{Z}_p^*$ this is $p-1$). If you get the complete group, it is a generator, otherwise it is not. In practice, groups are constructed in a way which allows to obtain a generator without trying all powers. A generator g of a group of integers modulo a prime p is called a primitive root.

Find all generators of $\mathbb{Z}_{11}^*$.

In the Diffie-Hellman key exchange, you have to raise a generator to large powers. This can be quite expensive if no tricks are used. Therefore, we introduce the fast exponentiation algorithm, also known as square and multiply algorithm. See Algorithm 1 for a description. All calculations are performed in the underlying monoid, e. g., if you use $\mathbb{Z}_p^*$ (which is a group and therefore also a monoid) you have to apply the modulo after each multiplication to keep the numbers small.

Calculate 42^{497} in $\mathbb{Z}_{1361}^*$ using fast exponentiation and a handheld calculator (not the one running on your computer or a programming language). Document a, e and n for each step.

There is little left to our small introduction. We will take a look at the ϕ function and at Fermat's little theorem and its generalization.

The ϕ function describes the group order of $\mathbb{Z}_n^*$, i. e., the number of elements which are invertible modulo n. For $n = pq$ with p, q prime, $\phi(n) = (p-1)(q-1)$. The more general case requires the complete factorization of n and will not be discussed here.

According to Fermat's little theorem, for a prime p and an integer a, it holds that $a^p = a \pmod{p}$. For $a \neq 0$ this can be turned into $a^{p-1} = 1 \pmod{p}$. A generalization is Euler's theorem, which says $a^{\phi(n)} = 1 \pmod{n}$ if $\gcd(a, n) = 1$.

[2] There is a better way if you can factor the group order, but that is usually not the case.

Input: Element a of a monoid and integer exponent n
Result: a^n (monoid element raised to the power of n)

Function `Fastexp`(a, n)**:**
  $e := 1$
  while $n > 0$ **do**
    if n odd **then**
      $e := a \cdot e$
    end
    $a := a^2$
    $n := \lfloor \frac{n}{2} \rfloor$
  end
  return e

Algorithm 1: Fast exponentiation algorithm

Task 2: The RSA cryptosystem

In this task we will take a deeper look at properties of the RSA cryptosystem. You will learn how to perform calculations in RSA and some pitfalls which can lead to vulnerabilities.

Your first task is to calculate the private key d from two primes p and q and to decrypt a ciphertext c with it. The primes are $p = 17$ and $q = 23$, the ciphertext is $c = 282$. The public key (N, e) consists of $N = 391$ and $e = 17$. To obtain the private key (N, d) you have to calculate the inverse d of e in $\mathbb{Z}^*_{\phi(N)}$, i. e., the d for which $e \cdot d \equiv 1 \pmod{\phi(n)}$. **Using that private key, decrypt** c.

The drunken developer Donald wants to encrypt his data with RSA. He reads up the API description of his favorite crypto library, which says, that `RSA.generateKey` requires a parameter called e. Since Donald does not know what e does, he just chooses $e = 1$. For the padding he has the choice between PKCS#1 and OAEP. He chooses OAEP. The library does not complain and the ciphertexts look random. **Did Donald choose reasonable parameters? Explain!**[3]

Find a value of e **for which RSA encryption is the identity function**, i. e. where $c = m$ for all m. Hint: Your e depends on n in some way and is not a fixed value.

Alice uses RSA with a 4096 bit key, i. e., the size of the modulus is $\approx 2^{4096}$. Her public exponent is $e = 3$. Bob encrypts a 256 bit AES key for Alice, without using padding. **How could the attacker Eve get the AES key without factoring?**[4]

Alice and Bob had a party at Alice's flat in Bamberg. Bob left his bicycle at Alice's place, locked with a combination lock (5 digits), because he preferred to go home via cab due to the high amount of snow. On the next day, Alice wants to visit Bob and bring him his bicycle. Since she does not have the combination for the lock, she asks Bob to send her a message with the combination in encrypted form, using her RSA public key. Instead of using OEAP, he invents his own padding scheme: Bob prepends the combination with repeating nines, until the message is the size of the RSA modulus (i. e., the number of digits of the message and the RSA modulus is the same) and encrypts it, using Alice's public key. **How could an adversary Eve get the combination for the lock if she intercepts the message?**

[3] Sadly, this is based on a true story.
[4] And without using a $5 wrench against Alice (https://xkcd.com/538/)