CS 5823/4823 Cryptography Midterm Project

Due March 28th, 2018, 11:59pm

This project intends to help you understand the concept of smoothness in cryptoanalysis. A monic polynomial $f(x)$ over a field is b-smooth if all its irreducible factors have degree less than or equal to b. For example, over $\mathbb{F}_2$, $x^{15}+x^3+1$ is 6-smooth since

$$(x^{15}+x^3+1) = (x^3+x+1) * (x^6+x^3+1) * (x^6+x^4+x^2+x+1).$$

An integer is s-smooth if all the prime factors are less than or equal to s. For example, 47711592 is 101-smooth since

$$47711592 = 2^3 * 3^{10} * 101.$$

You are allowed to work in a group of $g \leq 4$ students. Let p be

```
0xFFFFFFFF FFFFFFFF C90FDAA2 2168C234 C4C6628B 80DC1CD1
  29024E08 8A67CC74 020BBEA6 3B139B22 514A0879 8E3404DD
  EF9519B3 CD3A431B 302B0A6D F25F1437 4FE1356D 6D51C245
  E485B576 625E7EC6 F44C42E9 A637ED6B 0BFF5CB6 F406B7ED
  EE386BFB 5A899FA5 AE9F2411 7C4B1FE6 49286651 ECE65381
  FFFFFFFF FFFFFFFF,
```

which is a 1024-bit prime number used in Internet Key Exchange. (`https://tools.ietf.org/html/rfc7296#appendix-B.2`). A multiplicative generator for $\mathbb{F}_p$ is 2.

To complete the project, you or your group should submit an integer A and a sequence of integers $p_1, e_1, p_2, e_2, \cdots, p_l, e_l$ (all in the decimal representation) so that

1. $1 \leq A \leq 2^{1023}$.
2. $p_1 < p_2 < \cdots < p_l$ are prime numbers, and $e_1, e_2, \cdots, e_l$ are positive integers.
3. $p_1^{e_1} p_2^{e_2} \cdots p_l^{e_l} < p^2$
4. The most significant $9g$ decimal digits of A are concatenation of student ID numbers in the group.
5. It holds that
$$2^A = p_1^{e_1} p_2^{e_2} \cdots p_l^{e_l} \pmod{p} \quad (1)$$

You earn half of the credit if all of the above requirements hold. The other half will depend on how smooth the left-hand side of (1) is, namely, you should minimize p_l. The smaller the smoothness is, the higher points you will earn.

Note:

- Factorization is hard, but primes are abundant. More precisely, if you compute $R = 2^A \bmod p$, factoring R takes prohibitively long time, but if you remove all the small prime factors from R, there is a nontrivial probability that the remaining factor is a prime.
- The exponentiation is done on base 2, hence you may find a way to take advantage of the special forms.
- Sage/Python is slower than C/C++. You may want to use NTL/GMP for efficiency. Sage provides an interface to the NTL C++ library.

For the convenience of grading, please put your A, p_i's and e_i's in a Sage expression following the format of (1). Please submit your source code and a summary of running time and search space. Only one member in a group needs to submit. Please include the member names in the submission. Every member in the same group receives the same grade. No two groups can have overlapping memberships. Please be warned that if you decide to work in a group rather than work individually, you have a smaller search space.