CryptoHWRemarks

Krishna Acharya

January 2020

1 Q1

. All the codes written work, they have appropriate documentation, output with running time has been provided

2 Q2

2.1 Q2a

Takagi improved upon textbook RSA, and suggested using $N = pq^r$, Lets compare the cost.

- **Keygen** and **Encryption** (since we work with small public exponent) have the similar running time as textbook RSA.
- **MAIN IDEA:** Compute $d_p = e^{-1}(mod\ p-1)$ and $d_q = e^{-1}(mod\ q-1)$, get $m_p = c^{d_p}(modp)$ and $m_q = c^{d_q}(modq)$ [1]
  Hensel lift is used for $m(mod\ q^r)$. Given $m_i^e = c(mod\ q^i)$ we can lift solution to q^{i+1}. $m_{i+1}^e = (m_i + xq^i)^e = m_i^e + x(em_i^{e-1})q^i = c(mod\ q^{i+1})$. The starting lift value is just m_q computed earlier.
- Combine the two solutions using Chinese remainder theorem
- Why this is faster, requires $r-1$ lifts, and exponentiation for m_p and m_q,.
  for the similar bit size N, the corresponding p and q required to represent it are smaller, hence the exponentiation is cheaper.

2.2 Q2b

Boneh, Durfee and Howgrave-Graham in [1] come up with an algorithm that factorizes pq^r in time $\mathcal{O}\left(2^{k^{1-\epsilon}+\mathcal{O}(\log k)}\right)$. Where p and q are k bit primes [2] and $r = k^\epsilon$.

In particular $r = \Omega(\log p)$ is equivalent to $k = 1$ which results in the running time being $\mathcal{O}\left(2^{\mathcal{O}(\log k)}\right) = \mathcal{O}\left(poly(k)\right)$ i.e polynomial in the number of bits k
For their result they build the lattice with the vectors corresponding to $g_{i,k} := N^{m-k}x^i f^k(x)$ where $f(x) = (\bar{q}+x)^r$
There is a main lemma and Proof which I won't go into further detail here.

2.3 Q2c

I tried implementing the following algorithm from the same paper [1] [3] The attempt can be found in Q5.py

References

[1] Dan Boneh, Glenn Durfee, and Nick Howgrave-Graham. Factoring n = pr q for large r. 1999.

[1] Using CRT for decryption
[2] i.e $|\log p - \log q| \leq \mathcal{O}(1)$
[3] I tried to adapt their method by just using small roots in sage, but had no progress here