Pseudo Random Generator(PRG)

Sarthak Mahajan, 2018111005

March 7, 2022

1 Assumptions:

We assume that the discrete log Problem(DLP) is a one way function. The hc(hard core predicate) for DLP is the MSB(most significant bit) of the input. Discrete Logarithm Problem (in Z_p^*) is:

$$f_{p,g}(x) = g^x \bmod p$$

Thus single bit expansion PRG: $G_{single}(seed) = (g^x \bmod p, MSB(seed))$

2 Definition of PRG:

Let $\ell(\cdot)$ be a polynomial and let G be a deterministic polynomial-time algorithm such that for any input $s \in \{0,1\}^n$, algorithm G outputs a string of length $\ell(n)$. We say that G is a pseudorandom generator. if the following two conditions hold:

- **Expansion**: For every n it holds that $\ell(n) > n$.
- **Pseudorandomness**: For all probabilistic polynomial-time distinguishers D, there exists a negligible function negl such that:

  $$|\Pr[D(r) = 1] - \Pr[\dot{D}(G(s)) = 1]| \leq \text{negl}(n),$$

  where r is chosen uniformly at random from $\{0,1\}^{\ell(n)}$, the seed s is chosen uniformly at random from $\{0,1\}^n$, and the probabilities are taken over the random coins used by D and the choice of r and s. The function $\ell(\cdot)$ is called the expansion factor of G.

3 One Way Functions:

A function $f : \{0,1\}^* \to \{0,1\}^*$ is one-way if the following two conditions hold:

- **Easy to compute:**.There exists a polynomial-time algorithm M_f computing f; that is, $M_f(x) = f(x)$ for all x.

- **Hard to invert:** For every probabilistic polynomial-time algorithm $\mathcal{A}$, there exists a negligible function negl such that

$$\Pr\left[\text{Invert}_{\mathcal{A},f}(n) = 1\right] \leq \text{neg}(n).$$

4 Hard Core Predicate:

A function hc: $\{0,1\}^* \to \{0,1\}$ is a hard-core predicate of a function f if

- hc can be computed in polynomial time, and
- for every probabilistic polynomial-time algorithm $\mathcal{A}$ there exists a negligible function neg such that

$$\Pr_{x \leftarrow \{0,1\}^n}[\mathcal{A}(f(x)) = \text{hc}(x)] \leq \frac{1}{2} + \text{negl}(n)$$

where the probability. is taken over the uniform choice of x in $\{0,1\}^n$ and the random coin tosses of $\mathcal{A}$.

5 Single bit expansion PRG from One way Functions:

THEOREM: Let f be a one-way permutation and let he be a hardcore predicate of f. Then, $G(s) \stackrel{\text{def}}{=} (f(s), hc(s))$ constitutes a pseudorandom generator with expansion factor $\ell(n) = n + 1$.

6 Arbitrary expansion PRG from single bit expansion:

6.1 Theorem:

Assume that there exists a pseudorandom generator with expansion factor $\ell(n) = n+1$. Then for any polynomial $p()$, there erists a pseudorandom generator with expansion factor $(n) = p(n)$.

6.2 Method:

We use single bit expansion recursively, in each expansion, we take out the last bit as output and send the remaining l bits(assuming we started with l bits) to the next iteration of single bit expanding PRG. In this way we get k output bits, where k is the number of times we applied single bit expanding PRG. We then concatenate the obtained output bits to the right of the final output from the single bit expansion PRG(with the MSB of output being the last bit of the 1st iteration of single bit expansion PRG)

7 Provable security using PRG:

7.1 Idea:

We know that the One time pad is provably secure, however since the Size the keyspace is greater than equal to Size of message space its extremely costly. By the use of PRG's we can construct larger pseudo-random keys from smaller keys. Thus, for a key k, message m, size(k)¡size(m), we can use PRG G, to make G(k) = size(m), and use G(k) as the new key in one time pad, thus making it less costly.

7.2 Construction of Provably secure encryption scheme using PRG:

- **Gen:** on input 1^n, choose $k \leftarrow \{0,1\}^n$ uniformly at random and output it as the key.
- **Enc:** on input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^{\ell(n)}$, output the ciphertext
$$c := G(k) \oplus m.$$
- **Dec:** on input a key $k \in \{0,1\}^n$ and a ciphertext $c \in \{0,1\}^{\ell(n)}$, output the plaintext message
$$m := G(k) \oplus c.$$