The Security of all RSA and Discrete Log Bits

Johan Håstad[*] Mats Näslund[†]

Abstract

We study the security of individual bits in an RSA encrypted message $E_N(x)$. We show that given $E_N(x)$, predicting any single bit in x with only a non-negligible advantage over the trivial guessing strategy, is (through a polynomial time reduction) as hard as breaking RSA. Moreover, we prove that blocks of $O(\log\log N)$ bits of x are computationally indistinguishable from random bits. The results carry over to the Rabin encryption scheme.

Considering the discrete exponentiation function g^x modulo p, with probability $1 - o(1)$ over random choices of the prime p, the analog results are demonstrated. Finally, we prove that the bits of $ax + b$ modulo p give hard core predicates for any one-way function f.

1 Introduction

What is to be meant by a *secure cryptosystem*? There are rigorously defined notions, given by Goldwasser and Micali [14], such as *semantic security*; "whatever can be computed efficiently from the cryptotext should also be computable without it". Obtaining semantic security requires rather elaborate constructions, and we cannot in general hope to achieve this by simply applying a natural one-way function. In fact, any deterministic, public-key crypto system *must* leak some information. It is therefore important also to analyze the security of *specific* information concerning the plaintext. We here study the question of given the encrypted message $E(x)$, is it feasible to predict even a single bit of x? Now, "feasible" refers to the existence of probabilistic, polynomial time algorithms, and we cannot exclude the possibility of "guessing" a bit of x. What we can hope for is that this is essentially all you can do. With this in mind, as a successful adversary, we consider one who on average has a small advantage over the trivial guessing strategy.

We study the particular case when $E(x) = E_N(x)$ is RSA encryption. Here N is the product of two large primes, see [24]. RSA has been investigated from many different angles over the last 20 years, but still relatively little is known

[*] Department of Numerical Analysis and Computing Science, Royal Institute of Technology, SE-100 44 Stockholm, Sweden, `email:johanh@nada.kth.se`

[†] Ericsson Research, SE-164 80 Stockholm, Sweden, `email:mats.naslund@era.ericsson.se`. Work performed while being at the Royal Institute of Technology.

about the security. It is known that certain information such as (x/N), the Jacobi symbol of x, leaks through $E_N(x)$. For the specific issue of security for individual bits in x, this has so far only been proven to be true for the $O(\log\log N)$ least significant bits. Starting from a relatively weak result, in a sequence of papers, [15, 3, 29, 11, 26, 8], this was improved, ending with the final proof of "complete" security by Alexi, Chor, Goldreich, and Schnorr in [1]. There are also other known security results for certain predicates that are related to the individual bits of x, e.g. $\mathrm{half}_N(x) \triangleq 1$ if $x \geq (N+1)/2$, 0 otherwise, see [15] for instance.

For the other, internal bits, however, the best known result up until now states that they can cannot be computed with probability greater than 3/4. By using relations between $\mathrm{half}_N(x)$ and the individual bits of x, Ben-Or, Chor, and Shamir proved in [3], that the internal bits cannot be computed with probability of success exceeding 15/16. By a reduction to this proof, the result in [1] for the least significant bit, then improved the result to 3/4, still leaving a large gap to the desired 1/2-result.

In this paper we show the following:

Theorem. For any constant c and all sufficiently large n, unless RSA can be broken[1] in random polynomial time, no single bit of $E_N^{-1}(x)$ (where $\lceil \log N \rceil = n$) can be predicted with advantage[2] exceeding n^{-c}.

Moreover, distinguishing a block of $O(\log n)$ bits of x from random bits is also as hard as inverting RSA.

For a given function $E(x)$, the concept of bit-security is of course only meaningful when computing $E^{-1}(x)$ is assumed (or known) to be hard. Under such assumptions, there are a few cases where all individual bits are known to be secure. Assuming that factoring Blum-integers is hard, Håstad, Schrift, and Shamir proved in [16] that given g^x modulo N, where N is a Blum-integer, all bits of x are individually secure. Näslund showed in [20] that all bits in affine functions modulo a (not too small) prime, $x \mapsto ax + b$ modulo p, are secure given the information a, b, p, and $f(x)$ for *any* one-way function f. Our results here are achieved by extending and combining this work with the work in [1, 3, 10, 20].

The techniques can also be extended to show the analog results for other functions. The results carry over to the Rabin encryption function, $x \mapsto x^2$ modulo N. For a randomly chosen prime p, with high probability, the results also hold with respect to the discrete exponentiation function $x \mapsto g^x$ modulo p. That is, for almost all p, predicting a single bit (or distinguishing blocks of bits from random bits) is as hard as computing discrete logarithms. We also give explicit primes p for which it seems hard to get the same results using the methods currently at our disposal. Finally we also prove that the individual

[1] Here, "breaking" simply means retrieving the message x with non-negligible success probability. In particular, our result is not connected to issues such as the relationship between RSA and factoring, recently investigated in [5].

[2] We do not give credit to trivial advantage due to bias.

bits of hash-functions $ax + b$ modulo p give unpredictable predicates for any one-way function f even if p is quite small.

The paper is organized as follows. After first giving some notation in Section 2, we, in Section 3, review some techniques used in previous results. The bulk of the paper then proves the security results for all individual RSA bits. Section 4 generalizes some well-known sampling techniques. For technical reason, we divide the study into two cases; the internal bits are treated in Section 5 (which is the essentially new case) and then the most significant bits in Section 6. In Section 7 the simultaneous security of $O(\log n)$ bits is proven. Section 8 discusses the special case of the Rabin encryption scheme. In Section 9 we show that the techniques can be extended to prove security for the bits of the discrete logarithm, and we end by proving that the bits of $ax + b$ modulo p give unpredictable predicates in Section 10.

2 Preliminaries

The model of computation used is that of probabilistic Turing machines running in time poly(n) where n is the length of the input, pptm for short. In general, $\|y\|$ denotes the length of the binary string y. If S is a set, $\#S$ is the cardinality of S and by $x \in_{\mathcal{D}} S$ we mean an x chosen at random according to the distribution $\mathcal{D}$ on S, $\mathcal{U}$ denotes the uniform distribution. If $T \subset S$, then $\lambda_S(T) \triangleq \#T/\#S$ is the standard uniform measure. (When S is obvious from the context, we write $\lambda(T)$.) For two sets S, T, $S \nabla T$ is the *symmetric difference*: $(S \setminus T) \cup (T \setminus S)$.

We call a function $g(n)$ *negligible* if for every constant $c > 0$ and all sufficiently large n, $g(n) < n^{-c}$. A *one-way function* is a poly-time computable function f such that for every pptm, M, the probability that $M(f(x)) \in f^{-1}(f(x))$ is negligible. The probability is taken over $x \in_{\mathcal{U}} \{0,1\}^n$ and M's random coin flips.

Let f be a one-way function and let b be a poly-time computable boolean function. An $\epsilon(n)$-*oracle* for b is a pptm $\mathfrak{O}$ for which $\Pr[\mathfrak{O}(f(x)) = b(x)] \geq \frac{1+\epsilon(n)}{2}$, the probability taken over $x \in_{\mathcal{U}} \{0,1\}^n$, and $\mathfrak{O}$'s random choices. The only interesting case is when $\epsilon(n) > 0$. If no $\epsilon(n)$-oracle exists, we call b $\epsilon(n)$-*secure for* f, and if b is $\epsilon(n)$-secure for all non-negligible $\epsilon(n)$, we say that b *is secure for* f.

For $m, z \in \mathbb{Z}$, $m > 0$, we write $[z]_m \triangleq z$ modulo m and put $\mathrm{abs}_m(z) \triangleq \min\{[z]_m, m - [z]_m\}$. If for some $\delta \in [0,1]$, $\mathrm{abs}_m(z) \leq \delta m$, z is said to be δ-*small* (modulo m). A number x is δ-*determined* modulo m if it can be written on the form $y + z$ where y is known and z is δ-small. The gcd of $a, b \in \mathbb{Z}$ is written (a, b).

We use $E_N(x)$ to denote the RSA encryption function: $E_N(x) \triangleq [x^e]_N$ for $\|N\| = n$, $N = pq$, the product of two primes, and e, an integer relatively prime to $(p-1)(q-1)$.

For $z \in \mathbb{Z}$, $0 \leq i < \|z\|$, $\mathrm{bit}_i(z)$ denotes the ith bit in the binary representation of z, $\mathrm{bit}_i(z) \triangleq \lfloor z/2^i \rfloor$ modulo 2. This means that the bits are numbered $0, 1, \ldots, \|z\| - 1$, "right-to-left". In particular $\mathrm{lsb}(z) \triangleq \mathrm{bit}_0(z)$. For

$0 \leq i \leq j < \|z\|$, let $\mathrm{B}_i^j(z)$ denote bits $i, i+1, \ldots, j$ in the binary representation of z.

For a given N, and random z, the bits in $[z]_N$ are not uniformly distributed since the uniform distribution on $\mathbb{Z}_N$ is not the same as the uniform distribution on $\{0,1\}^{\|N\|}$. By the *bias* of the ith bit we mean the value $\beta_i(N)$ such that $\Pr_{z \in_{\mathcal{U}} \mathbb{Z}_N}[\mathrm{bit}_i(z) = 0] = \frac{1+\beta_i(N)}{2}$. It is an easy exercise to verify that always, $\beta_i(N) \leq \frac{2^i}{N}$. The bias is therefore only of significance for the $O(\log \log N)$ most significant bits. A notion of $\epsilon(n)$ security of biased bits is given in Section 6.

Finally, let $\mathcal{D}, \mathcal{D}'$ be distributions on the same space S. We call $\mathcal{D}, \mathcal{D}'$ (polynomially) *distinguishable* if there is a pptm D such that

$$\left| \Pr_{y \in_{\mathcal{D}} S}[D(y) = 1] - \Pr_{y' \in_{\mathcal{D}'} S}[D(y') = 1] \right|$$

is non-negligible.

A warning about convention. In many places we define integers by an expression that gives a real number. If the number is not integral we simply round it to one of the two closest integers. Sometimes we round explicitly i.e. by writing $\lfloor x \rfloor$ but at other times, for readability reasons, we do not.

3 Previous Work and Proof Outline

The security of the least significant bit in an RSA encrypted message has gained a lot of attention. The first result by Goldwasser, Micali, and Tong, [15], was to prove a $1 - o(1)$-security result. They used the relation $\mathrm{half}_N(x) = \mathrm{lsb}([2x]_N)$ (half_N as in the introduction), enabling a binary search to find x. By introducing a gcd computation technique a $\frac{1}{2} + o(1)$ result was given in [3] by Ben-Or, Chor, and Shamir. Further progress (still using the gcd technique) was accomplished by a more intricate sampling technique, and then by an improved combinatorial analysis of this technique. More precisely, Vazirani and Vazirani, [29], and then Goldreich, [11], respectively, showed 0.464- and 0.45-security. The main drawback of the method in [3] is that queries to the oracle are made in pairs, causing so called *error-doubling.*

By improving the sampling techniques once again, Schnorr and Alexi, [26], proved ϵ-security for any constant ϵ. They removed the error-doubling phenomenon by using "preprocessing". The cost of this preprocessing was, however, exponential in ϵ^{-1}.

To show $\epsilon(n)$-security for any non-negligible $\epsilon(\cdot)$, Chor and Goldreich managed in [8] (see also [1]) to reduce the cost of preprocessing to $\mathrm{poly}(\epsilon^{-1})$ by introducing the so called *two-point based sampling.* Recently, a simpler proof of $\epsilon(n)$-security was given in [10] by Fischlin and Schnorr. This last method does not use a gcd computation. Instead, the main idea is to use lsb-information to iteratively improve an approximation for the rational number $\frac{x}{N}$.

The results for the least significant bit generalizes in a straightforward way to any of the $O(\log n)$ least significant bits. For the internal bits of RSA however, the results so far are not very strong. The first appeared in the paper [15], where it was shown that for each i, there *are* N of very special form, for which the ith bit of x cannot be computed without errors. In [3], it was proved that an oracle for the ith bit of RSA can be converted into an lsb-oracle, increasing the error probability by $\frac{1}{4}$ in the worst case. However, they could also prove that for every second bit-position i, the error introduced could be bounded by $\frac{3}{16}$. Hence, from their own result for the lsb, a $\frac{7}{8}$-security for "half" of the individual bits followed. All later progress in proving security for the lsb has then, via the reduction by Ben-Or et al., strengthened the provable security for the internal bits. The best result so far is the $\frac{1}{2} + o(1)$-security that follows from the work in [1], still leaving a large gap to the desired $o(1)$ result. The provable security obtainable by these reductions depends on N and i (the bit-position considered), but for worst case N and i, results better than $\frac{1}{2} + o(1)$ are impossible by this "standard" reduction. If the oracle for the ith bit we start with is correct with probability $\frac{1+\epsilon'}{2}$, then after the conversion to an lsb-oracle, a success probability non-negligibly greater than $\frac{1}{2}$ must remain. The extra $\frac{1}{4}$ error that the reduction may add to the error probability is a tight bound, so we certainly need $\frac{1+\epsilon'}{2} - \frac{1}{4} > \frac{1}{2}$, i.e. $\epsilon' > \frac{1}{2}$.

As mentioned, few results of bit security for all individual bits in some function are known. In [20], it was claimed that *all* bits in functions of the form $x \mapsto [ax+b]_p$, p an $\Omega(n)$-bit prime, were $\epsilon(n)$-secure with respect to *any* one-way function. However, upon completing the proofs, it has become clear that the methods outlined there can not give better results than $\frac{3}{4}$-security for general p. In fact, it was this completion that led us to realize that the techniques apply to RSA as well. The common property between the two types of functions is multiplicativity; $E_N(cx) = [E_N(c)E_N(x)]_N$ and $[ch(x)]_p = [(ca)x + cb]_p$. That is, even if x is unknown, given $E_N(x)$ one can compute $E_N(cx)$, and given $h(x)$, $[ch(x)]_p$ can be found as $h'(x)$, another function of the same type. This property is used extensively in obtaining the previous RSA results and also in [20]. Of course, $h(\cdot)$ above has an extra feature; additive properties ($[h(x) - c]_p = [ax + (b-c)]_p$). However, it will be shown that we do not need that property.

In Section 10 we give the proofs of the results of [20] extended to allow primes of smaller size. This extension makes essential use of the results of Goldreich, Ron, and Sudan [13] giving an error correcting version of the Chinese remainder theorem.

Our proofs are by *reductio ad absurdum*: if an $\epsilon(n)$-oracle for $\text{bit}_i(x)$ exists, then this oracle can be used in a black-box fashion to retrieve x, i.e. to invert the one-way function we are currently considering .

3.1 The Method of Fischlin and Schnorr

To compute x using an lsb-oracle [10] proceeds as follows. Given is an initial guess y with $|y - x| < N/n^k$ for some k. Then by calculating $\text{lsb}(x)$ we get a guess, $(y - \text{lsb}(x))/2 + \text{lsb}(x)(N+1)/2$, of $x/2$ with half the uncertainty. Repeating this about n gives an exact value for a number of the form $x2^{-l}$ and from this we can retrieve x. Finally note that we can in advance specify a polynomial number of initial values of y one of which will be accurate enough.

It turns out that it is not necessary to have a very accurate lsb-oracle to start with to make this procedure work. Let an *interval* $J \subset [0..N-1]$ denote a set of consecutive integers in $\mathbb{Z}_N$ and for $z \in \mathbb{Z}$, $J + z$ is the interval J translated by z, allowing reductions modulo N. Suppose that for some not too short interval J, we have an oracle that, when given $E_N(z)$, is somewhat more likely to answer "1" for $z \in J$ than for $z \in J + (N+1)/2$. Now ask this oracle about $E_N([2^{-1}x]_N)$. We used above that

$$[2^{-1}x]_N = \frac{x - \mathrm{lsb}(x)}{2} + \mathrm{lsb}(x)[2^{-1}]_N = \frac{x - \mathrm{lsb}(x)}{2} + \mathrm{lsb}(x)\frac{N+1}{2}, \qquad (3.1)$$

see also Figure 1. Hence, if $\frac{x-\mathrm{lsb}(x)}{2} \in J$, then $[2^{-1}x]_N \in J + \mathrm{lsb}(x)(N+1)/2$. Since the oracle "behaves" differently on J, $J + (N+1)/2$, there is some hope to determine the lsb by querying the oracle.

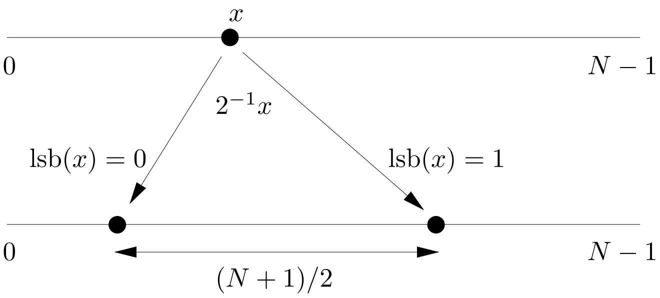

Figure 1: Division by 2 in $\mathbb{Z}_N$. Values that only differ in their lsb's are mapped to points $\frac{N+1}{2}$ apart.

There are some technical details that needs to be taken care of however. For instance, it is not clear how to get $\frac{x-\mathrm{lsb}(x)}{2}$ to lie in J in the first place, and a more serious concern is the existence of such J.

3.2 The Method of Näslund

Here the objective was to use an oracle for the ith bit in the function $x \mapsto [ax+b]_p$, p an $\Omega(\|x\|)$-bit prime and a, b random elements in $\mathbb{Z}_p$, to retrieve x.

To handle the internal bits, the main idea in [20] was to convert the oracle for the ith bit into an oracle that computed *both* the lsb *and* the $i+1$st bit, creating a two-bit window that by manipulating a, b through multiplications can be made to slide over all the bits in $[ax+b]_p$, see Figure 2.

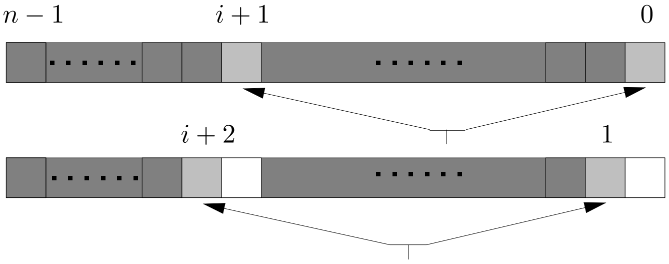

Figure 2: Deciding bits two-by-two.

As mentioned, a closer study of this work reveals that the methods in fact do not apply for some "highly structured" oracles that behave in a certain way. On the other hand, the oracles for which the methods fail are of a very special nature that we can exploit. We mentioned above that the tools from [3] can not be used to prove stronger security than $\frac{1}{2}$ for general N. The plan is now: (a) Investigate how, and when, the methods in [20] are applicable to prove bit security for RSA. (b) Show that when those methods fail, we can deduce that a certain relation between N and 2^{i+1} holds (i is the bit position predicted by the oracle), and furthermore, the oracle must then have a certain structure. (c) Prove that for bad N, i, and oracles as specified by (b), this makes it possible to construct an algorithm, i.e. a new oracle, $\mathfrak{O}'$, using the original oracle $\mathfrak{O}$ as a black box, such that $\mathfrak{O}'$ is an lsb-oracle. That is, *either* the methods from [20] works *or* the methods in [3] can be refined to prove the desired result.

We start by giving some generalizations of well-known sampling techniques and then formalize how the method by Fischlin and Schnorr is used as a "warm-up". We then follow (a), (b), (c) as above.

4 Sampling Techniques

Throughout the paper, i is reserved to denote the bit-position predicted by the oracle and $\epsilon(n)$ is reserved for the advantage of the oracle. More precisely, we assume that we have an oracle $\mathfrak{O}$ that given $E_N(x)$, $\|N\| = n$, predicts the ith bit of x with probability at least $\frac{1+\epsilon(n)}{2}$ where $\epsilon(n)$ is non-negligible.

Definition 4.1. By an *interval*, J, we mean a set of consecutive values $J = \{[u]_N, [u+1]_N, \ldots, [v]_N\}$ in $\mathbb{Z}_N$. The *length* of J is $\#J$ and the *measure* is $\lambda(J) \triangleq \#J/N$. If J is an interval and $z \in \mathbb{Z}_N$, denote by $J + z \triangleq \{[y+z]_N \mid y \in J\}$.

For a distribution $\mathcal{D}$ with support on $J \subset \mathbb{Z}_N$, let $P^{\mathfrak{O}}_{\mathcal{D}}(J)$ be the fraction of 1-answers the oracle gives on $\mathcal{D}$:

$$P^{\mathfrak{O}}_{\mathcal{D}}(J) \triangleq \mathrm{E}_{z \in_{\mathcal{D}} J}[\mathfrak{O}(E_N(z))] = \Pr_{z \in_{\mathcal{D}} J}[\mathfrak{O}(E_N(z)) = 1].$$

If $\mathcal{D}$ is the uniform distribution on J, we shall omit it from the notation, and furthermore, we then also define for $J_1, J_2 \subset \mathbb{Z}_N$: $\Delta^{\mathfrak{D}}(J_1, J_2) \triangleq \left|P^{\mathfrak{D}}(J_1) - P^{\mathfrak{D}}(J_2)\right|$.

Fix an $a \in \mathbb{Z}_N$ and suppose we have some set of random values $R = \{r_j\} \subset \mathbb{Z}_N$. Using the multiplicative properties of RSA, we can query the oracle for the ith bit of

$$R' = \{E_N^{-1}([(r_j + a)x]_N) \mid r_j \in R\}.$$

The idea is that in cases when some bit (or bits) of x equals 0, R' corresponds to some distribution $\mathcal{D}_0$ on $\mathbb{Z}_N$, and when the bit is 1, it corresponds to a distribution $\mathcal{D}_1$. If theses two distributions are polynomially distinguishable, we can by taking enough samples almost surely decide the relevant bit(s) of x in this way.

Now, these distributions $\mathcal{D}_0, \mathcal{D}_1$ have support on two subsets of $\mathbb{Z}_N$ (e.g. when we want to distinguish between values in some interval J and $J + (N + 1)/2$). To make sure that we hit one of these two subsets when sampling, we make sure that we know in advance the approximate locations in $\mathbb{Z}_N$ of the sample points. We will in fact later also need to know more than just the approximate locations, so we therefore state the following lemma.

Lemma 4.2. Let $m(n) \in \text{poly}(n)$, $d_I(n), d_Y(n) \in O(\log n)$. Then, given $E_N(x)$ and $r, s \in_{\mathcal{U}} \mathbb{Z}_N$, it is in deterministic polynomial time possible to generate a list of $m(n)$ values of the form $E_N(r_j x)$ so that each $[r_j x]_N$ is uniformly distributed and the values in $\{[r_j x]_N\}$ are pairwise independent. Furthermore, we generate a set consisting of $2^{4+2(d_I(n)+d_Y(n))} m(n)^2$ pairs of lists, $\{(L^I, L^Y)\}$, each L^I consisting of $m(n)$ values in $\mathbb{Z}_{2^{i+1}}$ and each L^Y of $m(n)$ values in $\mathbb{Z}$.

For at least one $(L', L'') \in \{(L^I, L^Y)\}$, for each $j = 1, \ldots, m(n)$, for some z_j so that $[z_j]_N = [r_j x]_N$ we have

$$\left|z_j - L''_j\right| \leq \frac{N}{2^{d_Y(n)}} \tag{4.1}$$

and

$$\text{abs}_{2^{i+1}}(z_j - L'_j) \leq 2^{i+1-d_I(n)}. \tag{4.2}$$

The reader is encouraged to compare this to §4.4 of [1]. There, it was only necessary to know the lsb of each point.

Proof. Let $U = [rx]_N$, $V = [sx]_N$, and $r_j = (r + js)$, $z_j = U + jV$, so that $[r_j x]_N = [z_j]_N$, $j = 1, \ldots, m(n)$. We easily see that this gives uniformly distributed values $[r_j x]_N \in \mathbb{Z}_N$ that are pairwise independent (see [7]). Repeat the following for all possibilities of

$$\mathrm{B}^i_{i-d_I(n)}(U), \qquad \mathrm{B}^i_{i-(d_I(n)+\log m(n))}(V), \tag{4.3}$$

and

$$u' \triangleq \frac{2^{1+d_Y(n)} U}{N}, \qquad v' \triangleq \frac{2^{(1+d_Y(n)+\log m(n))} V}{N}. \tag{4.4}$$

Notice that there are $2^{4+2d_I(n)+2d_Y(n)}m(n)^2$ possibilities all together. For each we create one (L^I, L^Y)-pair as described below. Let us focus on the one based on the correct values above.

Since by (4.4) above, we know V within $N/2^{d_Y(n)+1+\log m(n)}$ and $j \leq m(n)$, we know jV to within $N/2^{d_Y(n)+1}$. We also know U within $N/2^{d_Y(n)+1}$ so $U + jV$ (i.e. z_j) is known within $N/2^{d_Y(n)}$. Hence, this gives us a L''_j such that $\left|U + jV - L''_j\right| \leq N/2^{d_Y(n)}$.

Furthermore, we make the following observations. First, by (4.3), $[V]_{2^{i+1}}$ is known to within $2^{i+1-(d_I(n)+1+\log m(n))}$, and $j \leq m(n)$, so we also know $[jV]_{2^{i+1}}$ within $2^{i+1-(d_I(n)+1)}$. We have $[U]_{2^{i+1}}$ with the same accuracy, so $[U + jV]_{2^{i+1}}$ is known within $2^{i+1-d_I(n)}$. □

Note in particular that (4.1) implies that each $[r_j x]_N$ is $\frac{N}{2^{d_Y(n)}}$-determined. To start with, we will in fact only need (4.1) above, (4.2) will be useful later.

To be able to distinguish between two subsets of $\mathbb{Z}_N$ by observing how the oracle behaves, we must first know how the oracle ought to behave in the two cases.

Lemma 4.3. Let $J \subset \mathbb{Z}_N$ with $\lambda(J)$ non-negligible such that membership in J can be determined in polynomial time. Then, for any non-negligible $\epsilon'(n)$, and $K(n) \in \text{poly}(n)$, it is in probabilistic polynomial time possible to compute a value $\tilde{p}$ such that

$$\Pr[\left|P^{\mathfrak{O}}(J) - \tilde{p}\right| \geq \epsilon'(n)] \leq \frac{1}{K(n)}.$$

Proof. Let $m'(n) = \epsilon'(n)^{-2}\ln(4nK(n))$, and set $m(n) = 4\lambda(J)^{-1}m'(n)$. Pick randomly and independently $x_1, \ldots, x_{m(n)} \in \mathbb{Z}_N$. For each x_j, such that $x_j \in J$, query the oracle on $E_N(x_j)$ and compute $\tilde{p}$ as the fraction of 1-answers the oracle gives. Two applications of Chernoff bounds now establishes the lemma: first bound the probability that $\#(\{x_j\} \cap J)$ is small; then the probability that $\tilde{p}$ deviates too much from the expected value, $P^{\mathfrak{O}}(J)$. □

5 Security of Non Leftmost RSA Bits

In this section, we consider i such that

$$\tau(n) + 4\log\epsilon(n)^{-1} + \log n + 33 \leq i \leq n - 3\tau(n) - \log\epsilon(n)^{-1} - 7$$

where $\tau(n) \triangleq 34 + 5\log\epsilon(n)^{-1} + \log n$. We impose these restrictions on i for two reasons. First, we need at least a logarithmic number of bit positions to “the right” of the oracle to make the proof work. This does not matter, since the $O(\log n)$ least significant bits are covered by previous results. Secondly, for bit positions among the $O(\log n)$ most significant bits, the bias imposed by the binary representation of N may be non-negligible, and we handle these bits in Section 6.

5.1 RSA inversion, Method 1

The main technical lemma needed of this section is the following. It generalizes slightly lemmas from [3, 1, 10].

Lemma 5.1. If $\mathfrak{O}$ is such that for some interval J we have $\Delta^{\mathfrak{O}}(J, J+(N+1)/2) \geq \epsilon'(n)$, where $\lambda(J), \epsilon'(n)$ are non-negligible, then we can in random polynomial time construct an oracle, $\mathfrak{O}'$ such that for all $\frac{\lambda(J)\epsilon'(n)}{512n}$-determined $[ax]_N$, $\mathfrak{O}'$ determines $\mathrm{lsb}([ax]_N)$ with probability at least $1-\frac{1}{2n}$.

We will later see how to use such an oracle to find x in a straightforward way using the methods of [10].

We use $\mathfrak{O}$ as a black-box to build the new oracle as follows. Use Lemma 4.2 to get a set of random, pairwise independent values in $\mathbb{Z}_N$ of the form $\{[r_j x]_N\}$ for which we know their approximate locations in $\mathbb{Z}_N$, that is, we know L_j so that $\mathrm{abs}_N(r_j x - L_j)$ is small. Let us assume the hypothesis "$\mathrm{lsb}([ax]_N) = 0$". Then, if the hypothesis is correct, since $[ax]_N$ is $\lambda(J)\epsilon'(n)/(512n)$-determined and we have good approximations of the numbers $[r_j x]$, we can almost surely tell whether $[2^{-1}ax + r_j x]_N = [(2^{-1}a + r_j)x]_N$ is in J or not. If so, ask the oracle about this value and otherwise, disregard this $r_j x$. Since the length of J is not too short, we will ask the oracle on some non-negligible fraction of the points. Now, if the hypothesis is correct, these are almost all points in J. If, on the other hand, the hypothesis is wrong ($\mathrm{lsb}([ax]_N) = 1$) we will query the oracle on points in $J + (N+1)/2$ and by observing the oracle's behavior (the fraction of 1-answers) we should be able to tell the two cases apart. Let us turn to the formal argument.

Proof of Lemma 5.1. By Lemma 4.3 we can assume that we have $\tilde{p}_0, \tilde{p}_1$, approximations to $P^{\mathfrak{O}}(J)$, $P^{\mathfrak{O}}(J + (N+1)/2)$ respectively, within $\epsilon'(n)/4$. This can be made to hold with probability at least $1 - 1/(4n)$, and we assume for concreteness that $\tilde{p}_1 > \tilde{p}_0$.

Furthermore, assume that, we as described in Lemma 4.2, have generated R', a set of

$$m(n) = 512\lambda(J)^{-1}n\epsilon'(n)^{-2}$$

pairwise independent, uniformly distributed values of the form $r_j x$ with each $[r_j x]_N$ known within $2^{-d(n)}N$ for $d(n) = 9 + \log\epsilon'(n)^{-1} + \log\lambda(J)^{-1} + \log n$. Actually, there are a polynomial number of candidates to these approximate locations, but let us concentrate on the correct one—we can make one oracle $\mathfrak{O}'$ for each possibility, and we can exhaustively try them all.

Consider the set

$$R = \{[(2^{-1}a + r_j)x]_N \mid [r_j x]_N \in R'\}.$$

Assuming that $\mathrm{lsb}([ax]_N) = 0$, we can for each j compute an a_j such that $[a_j - (2^{-1}a + r_j)x]_N$ is $\frac{\lambda(J)\epsilon'(n)}{256n}$-small. If $a_j \in J$, we decide that $[(2^{-1}a + r_j)x]_N \in J$, and otherwise that it is not and remove it from R.

Definition 5.2. If $\mathrm{lsb}([ax]_N) = 0$ and $[(2^{-1}a+r_j)x]_N \in J$ while $[(2^{-1}a+r_j)x]_N$ is not put into R (or the other way around) we call $(2^{-1}a + r_j)x$ *misclassified.* The same notion applies to the case when $\mathrm{lsb}([ax]_N) = 1$ with J replaced by $J + (N + 1)/2$.

Not too many points are misclassified.

Claim 5.3. The expected number of misclassified points is bounded by

$$m(n)\epsilon'(n)\lambda(J)/(64n).$$

We postpone the proof of the claim.

Ask $\mathfrak{O}$ about all points of R. If the number of 1-answers is at least

$$m(n)\lambda(J)(\tilde{p}_0 + \tilde{p}_1)/2,$$

guess $\mathrm{lsb}([ax]_N) = 1$ and otherwise guess $\mathrm{lsb}([ax]_N) = 0$.

Let us estimate the probability of an incorrect answer. We assume that $\mathrm{lsb}([ax]_N) = 0$, the other case being similar. Let us analyze what would have happened if all points had been correctly classified. Note that in this case all points are uniformly distributed and pairwise independent. The expected number of points put into R and given the answer 1 is $P^{\mathfrak{O}}(J)\lambda(J)m(n)$ and the variance on this number is at most $P^{\mathfrak{O}}(J)\lambda(J)m(n)$. The probability that more than $\lambda(J)m(n)(P^{\mathfrak{O}}(J) + \epsilon'(n)/8)$ points are put into R and given the answer 1 is bounded, by Chebychev's inequality, by

$$\frac{64\lambda(J)P^{\mathfrak{O}}(J)m(n)}{\epsilon'(n)^2\lambda(J)^2m(n)^2} \leq \frac{64}{\epsilon'(n)^2\lambda(J)m(n)} \leq \frac{1}{8n},$$

where the last inequality follows from the definition of m. Now, unless at least $\lambda(J)m(n)\epsilon'(n)/8$ numbers are misclassified the number of 1-answers is, in the above case, bounded by $\lambda(J)m(n)(P^{\mathfrak{O}}(J) + \epsilon'(n)/4)$. By assumption,

$$\tilde{p}_0 \geq P^{\mathfrak{O}}(J) - \epsilon'(n)/4$$

and

$$\tilde{p}_1 \geq P^{\mathfrak{O}}(J + (N + 1)/2) - \epsilon'(n)/4 \geq P^{\mathfrak{O}}(J) + 3\epsilon'(n)/4$$

and thus $P^{\mathfrak{O}}(J)+\epsilon'(n)/4 \leq (\tilde{p}_0+\tilde{p}_1)/2$ and hence in the above case the algorithm would output the correct answer. Since, by Claim 5.3 the probability of having $\lambda(J)m(n)\epsilon'(n)/8$ misclassified points is bounded by $1/(8n)$ adding the failure probabilities, the lemma follows. □

It remains to prove Claim 5.3

Proof of Claim 5.3. Since the points in question are $\epsilon'(n)\lambda(J)/(256n)$-determined the only points that can be misclassified are those which are within at most this distance of either endpoint of J. Since the points are uniformly distributed the expected number of such points is $m(n)\epsilon'(n)\lambda(J)/(64n)$. □

Let us see how to use Lemma 5.1 to invert RSA.

Lemma 5.4. If $\mathfrak{O}$ is such that for some interval J we have $\Delta^{\mathfrak{O}}(J, J + (N + 1)/2) \geq \epsilon'(n)$, where $\lambda(J), \epsilon'(n)$ are non-negligible, then we can, in random polynomial time, recover x with probability at least $1/2$.

Proof. Given the oracle $\mathfrak{O}'$ proved to exist by Lemma 5.1 we proceed as follows with all arithmetic modulo N.

Algorithm 5.5.

```
Input: $E_N(x) = [x^e]_N$, $\|N\| = n$
Output: $x$
(1)  guess $y$ so that $\mathrm{abs}_N(x - y) \leq N\lambda(J)\epsilon'(n)/512n$
(2)  $z \leftarrow E_N(x)$
(3)  for $j := 0$ to $n - 1$ do
(4)      $b \leftarrow \mathfrak{O}'(z, y)$
(5)      $z \leftarrow 2^{-e}z$;
(6)      $y \leftarrow b(N + 1)/2 + (y - b)/2$;
(7)  return $y2^n$
```

A sufficiently dense set of possible values of y can be tried in polynomial time and thus "guessing" is in fact replaced by a polynomially bounded loop. By induction, provided that all the oracle calls are answered correctly, y is at the call to $\mathfrak{O}'$ for a particular value of the loop variable j, an approximation of $2^{-j}x$ within $2^{-j}N\lambda(J)\epsilon'(n)/512n$ and z is the encryption of $2^{-j}x$. This implies that the preconditions of the parameters sent to the oracle remains correct and with probability at least $1 - n \cdot \frac{1}{2n} = 1/2$ we get n correct answers from the oracle. This implies that at the end of the algorithm y is in fact exactly $2^{-n}x$ and the algorithm is correct. □

We next to proceed to describe an alternate way to use an oracle to predict RSA. It is much more correlated directly with the i'th bit and hence more directly applicable to proving our main result.

5.2 RSA inversion, Method 2

This second method is much more technical than the previous, and we start by outlining the ideas. This method follows the principles used in [20].

The idea is to use the oracle for the ith bit to decide *both* the lsb *and* the $i + 1$st bit. Suppose that we already know the value of $\mathrm{B}_{i-d+1}^{i}(x)$, the value of the d bits to the right of, and including bit i. (If d is small enough we can initially simply guess this value.) As described in Section 3.2 the most intuitive approach would be to ask the oracle on $E_N([2^{-1}x]_N)$. For technical reasons we will, however, use $E_N([2^{-\tau}x]_N)$ where $1 < \tau \ll i$. Why $\tau > 1$ is a good idea is explained shortly. Make a list of all $2^{2\tau}$ possibilities for bits $i + 1, \ldots, i + \tau$, and bits $0, \ldots, \tau - 1$ in x, i.e, for $\mathrm{B}_{i+1}^{i+\tau}(x)$ and $\mathrm{B}_0^{\tau-1}(x)$. Hence, an entry in this list looks like (u_j, v_j), $0 \leq u_j, v_j \leq 2^\tau - 1$, u_j corresponding to a possibility for

$\mathrm{B}_{i+1}^{i+\tau}(x)$ and v_j to a possibility for $\mathrm{B}_0^{\tau-1}(x)$. The two bits we are after, $\mathrm{bit}_{i+1}(x)$ and $\mathrm{lsb}(x)$, then corresponds to $\mathrm{lsb}(u_j)$ and $\mathrm{lsb}(v_j)$, respectively.

Take any two distinct candidates from the list (u_1, v_1) and (u_2, v_2). Surely, they cannot both be correct, so we shall try to exclude one of them (the incorrect one if one is correct). Furthermore, since we only aim to determine the two bits $\mathrm{bit}_{i+1}(x), \mathrm{lsb}(x)$, we are only interested in pairs (u_1, v_1), (u_2, v_2) for which $\mathrm{lsb}(u_1) \neq \mathrm{lsb}(u_2)$ or $\mathrm{lsb}(v_1) \neq \mathrm{lsb}(v_2)$.

Now consider $[2^{-\tau}x]_N$.

$$\begin{aligned}[2^{-\tau}x]_N = {} & \frac{x - \mathrm{B}_{i+1}^{i+\tau}(x)2^{i+1} - \mathrm{B}_{i-d+1}^{i}(x)2^{i-d+1} - \mathrm{B}_0^{\tau-1}(x)}{2^\tau} \\ & + \mathrm{B}_{i+1}^{i+\tau}(x)2^{i+1-\tau} + \mathrm{B}_{i-d+1}^{i}(x)2^{i-d+1-\tau} \\ & + \mathrm{B}_0^{\tau-1}(x)[2^{-\tau}]_N. \end{aligned} \tag{5.1}$$

The term $x - \mathrm{B}_{i+1}^{i+\tau}(x)2^{i+1} - \mathrm{B}_{i-d+1}^{i}(x)2^{i-d+1} - \mathrm{B}_0^{\tau-1}(x)$ is divisible (as an integer) by 2^τ, and it has d zeros to the right of bit i, so it is very small modulo 2^{i+1}. Hence, $\mathrm{B}_{i+1}^{i+\tau}(x)2^{i+1-\tau} + \mathrm{B}_0^{\tau-1}(x)[2^{-\tau}]_N$ is essentially the only unknown term that influences the ith bit in $[2^{-\tau}x]_N$.

Now let us try to decide if $(\mathrm{B}_{i+1}^{i+\tau}(x), \mathrm{B}_0^{\tau-1}(x)) = (u_1, v_1)$ or (u_2, v_2), i.e. we would like to tell if $[2^{-\tau}x]_N$ is of the form $z' + u_1 2^{i+1-\tau} + v_1[2^{-\tau}]_N$ or of the form $z' + u_2 2^{i+1-\tau} + v_2[2^{-\tau}]_N$, and this is the same as distinguishing between values of the form z and $z + u2^{i+1-\tau} + v[2^{-\tau}]_N$, where $z = z' + u_1 2^{i+1-\tau} + v_1[2^{-\tau}]_N$, $u = u_2 - u_1$, and $v = v_2 - v_1$. Since are only interested in the differences, we may interchange (u_1, v_1) and (u_2, v_2) to ensure that $v \geq 0$. Because at least one of the pairs u_1, u_2 and v_1, v_2 differs in their least significant bit, we know that at least one of u, v is odd.

If we assume that z belongs to some subset $S \subset \mathbb{Z}_N$, then $[2^{-\tau}x]_N \in S$ if (u_1, v_1) is correct and $[2^{-\tau}x]_N \in S + u2^{i+1-\tau} + v[2^{-\tau}]_N$ if (u_2, v_2) is correct. We now make the following definition:

Definition 5.6. For given N, τ and $0 \leq v \leq 2^\tau - 1$, $|u| \leq 2^\tau - 1$, define

$$\alpha_N^\tau(u, v) \triangleq u2^{i+1-\tau} + v[2^{-\tau}]_N.$$

Note that $\alpha_N^\tau(u, v)$ is computed modulo N, not modulo 2^{i+1}. Again, we emphasize that we are only interested in $\alpha_N^\tau(u, v)$ where at least one of u, v is odd.

Just like we in the previous section wanted to find sets J, $J + (N+1)/2 = J + [2^{-1}]_N$, where the oracle behaved differently, we can now ask if there are similar sets $S, S + \alpha_N^\tau(u, v)$ where the oracle behaves differently. Consider first the case when v is odd. There are 2^τ distinct values of the form $k\alpha_N^\tau(u, v)$, $k = 0, 1, \ldots, 2^\tau - 1$, and one can hope that for at least one of these k's, the oracle distinguishes between some $S + k\alpha_N^\tau(u, v)$ and $S + (k+1)\alpha_N^\tau(u, v)$. When $k = 2^\tau$, $[k\alpha_N^\tau(u, v)]_N = u2^{i+1} + v$, which in turn is v modulo 2^{i+1}. Since v is small and the oracle predicts the ith bit, as far as the oracle is concerned, we are then essentially back where we started. When $\tau = 1$ there are therefore essentially only two possible multiples of $\alpha_N^1(u,v)$ and this is the reason why we use $\tau > 1$. Now, if we can find good interval pairs for *all* these α-values, we seem to be in good shape.

Consider a particular (u,v) and fix $S \subset \mathbb{Z}_N$ so that all $z \in S$ have the same value for their ith bit. We can thus not let S be an interval as before, since the length of S would then be bounded by 2^i, which is negligible compared to N. Instead, we take S as a union of short intervals, each at distance 2^{i+1}, i.e. $S = \bigcup_l (J' + l2^{i+1})$ where J' is a "traditional" interval of length at most 2^i and the range of l is chosen suitably so that the measure of the set S is non-negligible.

Definition 5.7. In the sequel we write N as $N \triangleq N_1 2^{i+1} + N_0$ where $N_0 < 2^{i+1}$. We sometimes also study N_1 closer, and it will be convenient to write N_1 as $N_1 \triangleq N_3 2^{\tau(n)} + N_2$ where $N_2 < 2^{\tau(n)}$.

Definition 5.8. Let $I \triangleq \mathbb{Z}_{2^{i+1}} = \{0, 1, \ldots, 2^{i+1} - 1\}$ and $Y \triangleq \mathbb{Z}_{N_1+1} = \{0, 1, \ldots, N_1\}$. We can view $\mathbb{Z}_N$ as a subset of $I \times Y$ by defining the natural projection $\pi : \mathbb{Z}_N \to I \times Y$ by

$$\pi(z) = (\pi_I(z), \pi_Y(z)) \triangleq (z \bmod 2^{i+1}, \lfloor z/2^{i+1} \rfloor).$$

Note that π is surjective, except for some values of the form (j, N_1) with $j \geq N_0$. We would like to draw the readers attention to the fact that since we are really working modulo N, the value z that $\pi(\cdot)$ is applied to should, when necessary, first be reduced modulo N. Such modular reductions could cause problems. For this reason, we mostly, but not always, arrange things so that the argument z (even when z is the sum of elements in $\mathbb{Z}_N$) can be considered as an integer in the range $[0..N-1]$. We define the *plane* $\Pi(N,i) = (I \times Y) \cap \pi(\mathbb{Z}_N)$. For $b \in \{0,1\}$ we set

$$S^{(b)} \triangleq \{z \in \mathbb{Z}_N \mid \mathrm{bit}_i(z) = b\}.$$

For all non-negative integers we define a *box*, S, of *width* w and *height* h as the following rectilinear subset of $I \times Y$:

$$\{\pi(z + 2^{i+1}y) \mid z_0 \leq z < z_0 + w,\ y_0 \leq y < y_0 + h\}.$$

The *measure* of such a box is simply $\lambda(S) \triangleq \frac{\#S}{N} = \frac{wh}{N}$ provided that $h < N_1$ and $w \leq 2^{i+1}$. Furthermore, for a box S and $z \in \mathbb{Z}_N$ we define the *z-translation* of S as

$$S + z = S + (\pi_I(z), \pi_Y(z)) \triangleq \{(\pi_I(z' + z), \pi_Y(y' + z)) \mid (z', y') \in S\}.$$

A *level* is a subset of $\Pi(N,i)$ consisting of the set of values having a fixed π_Y-value. All levels except possibly the N_1th level are of size 2^{i+1}.

Finally, if S is a box and $\mathcal{D}$ is a probability distribution on S, we define as before

$$P_{\mathcal{D}}^{\mathfrak{O}}(S) \triangleq \Pr_{z \in_{\mathcal{D}} S}[\mathfrak{O}(E_N(z)) = 1].$$

When $\mathcal{D}$ is the uniform distribution, we omit it from the notation and then also define $\Delta^{\mathfrak{O}}(S, S') \triangleq \left|P^{\mathfrak{O}}(S) - P^{\mathfrak{O}}(S')\right|$.

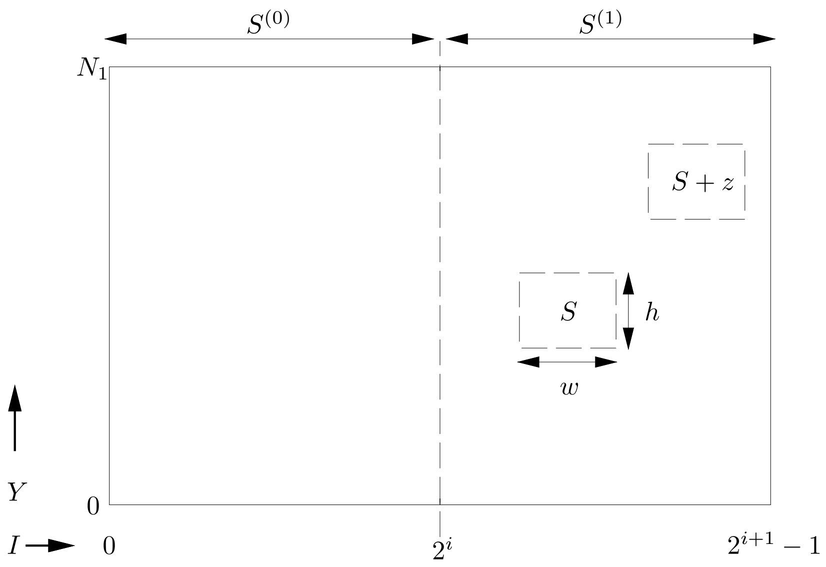

Figure 3: The $\Pi(N,i)$-plane. Shown is a typical box, S, and a translation, $S+z$.

Figure 3 below illustrates the plane.

In the figure, the relative scale on the I and Y-axis suggests that $i > n/2$, since $2^{i+1} = \#I > \#Y = \lceil N/2^{i+1} \rceil \approx 2^{n-(i+1)}$.

We now state the main lemma of this section.

Lemma 5.9. Suppose that for all $0 \leq v \leq 2^{\tau(n)} - 1$, $|u| \leq 2^{\tau(n)} - 1$, u or v odd, there is a box $S_{u,v}$ of width at least $w(n)2^{i+1}$, height at least $h(n)N_1$, and with $\Delta^{\mathfrak{O}}(S_{u,v}, S_{u,v} + \alpha_N^{\tau(n)}(u,v)) \geq \epsilon'(n)$, where $h(n), w(n), \epsilon'(n)$ are all non-negligible. Define

$$d(n) \triangleq \log \epsilon'(n)^{-1} + \log(w(n)h(n))^{-1} + 9 + 2\tau(n) + \log n.$$

Then it is possible to construct an oracle, $\mathfrak{O}'$, that given $E_N(x)$, j, $\mathrm{B}_{i-d(n)+1}^{i+j}(x)$, $\mathrm{B}_0^{j-1}(x)$, and y so that $\mathrm{abs}_N(x-y) \leq 2^{-d(n)}N$, for any $0 \leq j \leq \max(n-i-2, i)$, determines $\mathrm{bit}_{i+j+1}(x)$ *and* $\mathrm{bit}_j(x)$ with probability at least $1 - \frac{1}{2n}$.

Proof. We assume that, in fact, $j \leq \min\{i - d(n) + 1, n - d(n) - 1\}$. Otherwise, only one of the two bits $\mathrm{bit}_{i+j+1}(x), \mathrm{bit}_j(x)$ is unknown, and it is easy to see how that would only simplify the procedure below.

We define $\lambda_{u,v} \triangleq \lambda(S_{u,v})$ and

$$m(n) \triangleq 512n\lambda_{u,v}^{-1}\epsilon'(n)^{-2}2^{2\tau(n)}.$$

Let $\tilde{p}_{u,v}$ and $\tilde{p}'_{u,v}$ be estimates for $P^{\mathfrak{O}}(S_{u,v})$ and $P^{\mathfrak{O}}(S_{u,v}+\alpha_N^{\tau(n)}(u,v))$ respectively such that $|\tilde{p}_{u,v}-P^{\mathfrak{O}}(S_{u,v})| \leq \epsilon(n)'/8$ and $|\tilde{p}'_{u,v}-P^{\mathfrak{O}}(S_{u,v}+\alpha_N^{\tau(n)}(u,v))| \leq \epsilon(n)'/8$ with probability $1-1/(8n)$. Assume for notational simplicity that we always have $\tilde{p}'_{u,v} > \tilde{p}_{u,v}$.

By Lemma 4.2, we can generate $m(n)$ sample points of the form $r_k x$ where for some z_k, $[r_k x]_N = [z_k]_N$, z_k is known within $2^{-d(n)}N$ and with $[z_k]_{2^{i+1}}$ known with a relative error of $2^{-d(n)}$. There are a polynomial number of possibilities for these values but we can construct one oracle for each, and try them all, so we may assume that we have the correct choice.

The procedure to decide two new bits in x is:

Algorithm 5.10.

```
Output: $(\mathrm{bit}_{i+j+1}(x), \mathrm{bit}_j(x))$
(1)  $T \leftarrow \{0,1\}^{\tau(n)} \times \{0,1\}^{\tau(n)}$
(2)  while $\exists\,(u_1,v_1),(u_2,v_2) \in T$ s.t. $\mathrm{lsb}(u_1) \neq \mathrm{lsb}(u_2)$ OR $\mathrm{lsb}(v_1) \neq \mathrm{lsb}(v_2)$ do
(3)       possibly exchange $(u_1,v_1)$, $(u_2,v_2)$ to ensure $v_2 \geq v_1$
(4)       $(u,v) \leftarrow (u_2-u_1, v_2-v_1)$; $\alpha \leftarrow \alpha_N^{\tau(n)}(u,v)$
(5)       guess that $u_1 = \mathrm{B}_{i+j+1}^{i+j+\tau(n)}(x)$ and $v_1 = \mathrm{B}_j^{j+\tau(n)-1}(x)$
(6)       $R = \{\}$
(7)       for $k := 1$ to $m(n)$ do
(8)            $\pi' \leftarrow$ approximation to $\pi([(r_k+2^{-(j+\tau(n))})x]_N)$
                    based on $j, \tau(n), u_1, v_1$ and available info. on $x, r_k x$
(9)            if $\pi' \in S_{u,v}$ then
(10)             $R \leftarrow R \cup \{E_N((r_k+2^{-(j+\tau(n))})x)\}$
(11)      $p \leftarrow$ number of 1 answers of $\mathfrak{O}$ on $R$
(12)      if $p \leq \lambda_{u,v} m(n)(\tilde{p}_{u,v}+\tilde{p}'_{u,v})/2$ then
(13)         delete $(u_2,v_2)$ from $T$
(14)      else
(15)         delete $(u_1,v_1)$ from $T$
(16) pick any $(u,v) \in T$; return $(\mathrm{lsb}(u), \mathrm{lsb}(v))$
```

Some comments may be in place. The while-loop runs over pairs of candidates for $\mathrm{B}_{i+j+1}^{i+j+\tau(n)}(x)$, $\mathrm{B}_j^{j+\tau(n)-1}(x)$, and terminates when all remaining pairs have the same value both for $\mathrm{lsb}(\mathrm{B}_{i+j+1}^{i+j+\tau(n)}(x))$ (corresponding to $\mathrm{bit}_{i+j+1}(x)$) and $\mathrm{lsb}(\mathrm{B}_j^{j+\tau(n)-1}(x))$ (i.e. $\mathrm{bit}_j(x)$), meaning that we hopefully have decided two new bits in x.

In line (5) we "guess" that (u_1,v_1) is the correct choice for the unknown bits. This means that the computations that follow are made as if (u_1,v_1) is correct. The guess is needed to perform the computation in line (8). If (u_1,v_1) indeed is correct, then the π'-value computed are good approximations to the true π-values. Therefore, the distribution on the set R is close to uniform over $S_{u,v}$ and pairwise independent. We, similarly to the proof of Lemma 5.1 call a point *misclassified* if the decision whether to put it into R is incorrect. If instead, (u_2,v_2) is correct, then R consists of values close to the uniform distribution on $S_{u,v}+\alpha_N^{\tau(n)}(u,v)$ and we have a similar notion of misclassified.

Of course, we may be totally wrong so that neither (u_1, v_1) nor (u_2, v_2) is correct, but if so, we always (and correctly) rule out one of them as a possibility and there is nothing to analyze.

Thus assuming that either (u_1, v_1) or (u_2, v_2) is correct, let us analyze the probability of erroneously deleting the correct value in a single iteration. We claim the following (c.f. Claim 5.3).

Claim 5.11. The expected number of misclassified points is bounded by

$$\epsilon'(n)\lambda_{u,v}m(n)2^{-(2\tau(n)+6)}/n.$$

We postpone the proof of the claim.

Assume for concreteness that (u_1, v_1) is the correct value, the other case being similar. If no misclassifications were made, since the points are pairwise independent and uniformly distributed, the expected number of 1-answers is $P^{\mathfrak{O}}(S_{u,v})\lambda_{u,v}m(n)$ and the variance of this number is at most $P^{\mathfrak{O}}(S_{u,v})\lambda_{u,v}m(n)$. Thus, by Chebychev's inequality, the probability that more than

$$\lambda_{u,v}m(n)(P^{\mathfrak{O}}(S_{u,v}) + \epsilon'(n)/8)$$

1-answers are given is bounded by

$$\frac{64P^{\mathfrak{O}}(S_{u,v})\lambda_{u,v}m(n)}{\epsilon'(n)^2\lambda_{u,v}^2 m(n)^2} \leq \frac{64}{\epsilon'(n)^2\lambda_{u,v}m(n)} \leq \frac{2^{-2\tau(n)}}{8n}.$$

Thus unless $\epsilon'(n)\lambda_{u,v}m(n)/8$ points are misclassified the number of 1-answers obtained is in this case at most $\lambda_{u,v}m(n)(P^{\mathfrak{O}}(S_{u,v}) + \epsilon'(n)/4)$. By the assumption on $\tilde{p}_{u,v}$ and $\tilde{p}'_{u,v}$ we have

$$\tilde{p}_{u,v} \geq P^{\mathfrak{O}}(S_{u,v}) - \epsilon'(n)/4$$

and

$$\tilde{p}'_{u,v} \geq P^{\mathfrak{O}}(S_{u,v} + \alpha_N^{\tau(n)}(u, v)) - \epsilon'(n)/4 \geq P^{\mathfrak{O}}(S_{u,v}) + 3\epsilon'(n)/4.$$

These inequalities imply $P^{\mathfrak{O}}(S_{u,v}) + \epsilon'(n)/4 \leq (\tilde{p}_{u,v} + \tilde{p}'_{u,v})/2$ and thus we conclude we do not discard the correct value in this case.

Finally, by Claim 5.11, we conclude that the probability of having more than $\epsilon'(n)\lambda_{u,v}m(n)/8$ misclassified points is bounded by $2^{-2\tau(n)}/(8n)$. This implies that the probability of an error in one iteration is bounded by $2^{-2\tau(n)}/(2n)$ and since we have at most $2^{2\tau(n)}$ iterations, the lemma follows. □

We have to give the above postponed proof of Claim 5.11

Proof of Claim 5.11. A point can only be misclassified if it is close to the borders of S. In particular it should either be within distance $N_1 2^{-d(n)}$ in the Y-direction or within $2^{i+1-d(n)}$ in the I-direction. Since the points are uniformly distributed the expected number of such points is at most $m(n)2^{3-d(n)}$ and the claim follows by the definition of $d(n)$. □

Given the hypothesis of Lemma 5.9 it is not difficult to invert RSA.

Lemma 5.12. Given the same assumptions as Lemma 5.9, we can invert RSA in random polynomial time with probability of success at least $\frac{1}{2}$.

Proof. Apply Lemma 5.9 and get the resulting oracle $\mathfrak{O}'$. The inversion algorithm is now very simple.

Algorithm 5.13.

```
Input: $E_N(x)$, $\|N\| = n$
Output: $x$
guess $y$ so that $\mathrm{abs}_N(x-y) \le 2^{-d(n)}N$
guess $z' = \mathrm{B}^{i}_{i-d(n)+1}(x)$; $z \leftarrow 0$          /* $z = \mathrm{B}_0^{j-1}(x)$ */
for $j := 0$ to $\max(n-i-2, i)$ do
    $(b', b) \leftarrow \mathfrak{O}'(E_N(x), j, z', z, y)$        /* $\mathrm{bit}_{i+1+j}(x), \mathrm{bit}_j(x)$ */
    $z' \leftarrow 2^{j+d(n)}b' + z'$;                             /* $\mathrm{B}^{i+j}_{i-d(n)+1}(x)$ */
    $z \leftarrow 2^j b + z$;                                      /* $\mathrm{B}^j_0(x)$ */
return $z'2^{i+1-d(n)} + z$
```

We can repeat the process for all the polynomially many choices for y, z', so we may assume that we have a correct guess. If the oracle does not err, it is easy to see that the final $z'2^{i+1-d(n)} + z$ is the correct binary representation of x. Since $\mathfrak{O}'$ is used at most n times, the total error probability is at most $n\frac{1}{2n} = \frac{1}{2}$. □

The key to the overall proof is thus to establish the existence of the boxes needed for Lemma 5.12 or the interval needed for Lemma 5.4. This is the topic of the next section.

Before continuing let us, however, explain one point. We do not only need the existence of the given boxes/intervals but also that they can be found efficiently. Most of our proofs will in fact be efficient in this sense, but this is really not needed. If S is a good box of non-negligible size then so is any other box sufficiently close to S. It is not hard to see that once we have non-negligible lower bounds for the size and the advantage then we can in fact specify a polynomial number of candidates $\{S_j\}$ such that if a good box exists then in fact one of the S_j is also good, but of slightly inferior quality. This S_j can then be located by Lemma 4.3. This implies that existence is equivalent to efficiently being able to find a desired object and hence we can safely ignore this point.

5.3 Proving existence of good boxes/intervals

The main approach is to establish the existence of the boxes needed for Lemma 5.12. The analysis is divided into a number of cases and only in one case may we fail to directly establish the existence of the relevant boxes. In that case we prove that either the desired boxes exist, or, we can construct the interval needed for Lemma 5.4. We start with a simple case.

Lemma 5.14. If v is even and u is odd we have a $k \leq 2^{\tau(n)} - 1$ such that

$$\Delta^{\mathfrak{D}}(S^{(0)} + k\alpha_N^{\tau(n)}(u,v), S^{(0)} + (k+1)\alpha_N^{\tau(n)}(u,v)) \geq \epsilon(n)2^{-\tau(n)}.$$

We give the simple proof in Section 5.3.1 on page 21.

Odd v require a bit more careful analysis and we start by a definition of a new quantity that is intimately related to $\alpha_N^{\tau(n)}(u,v)$.

Definition 5.15. For $0 < v \leq 2^\tau - 1$, v odd, and $|u| \leq 2^\tau - 1$, define

$$\tilde{\alpha}_N^\tau(u,v) \triangleq [-uv^{-1}N]_{2^\tau} 2^{i+1-\tau} + \left\lceil \frac{N}{2^\tau} \right\rceil.$$

The key relation between $\alpha_N^{\tau(n)}(u,v)$ and $\tilde{\alpha}_N^{\tau(n)}(u,v)$ is given by the lemma below.

Lemma 5.16. Let v be odd. If there is a box S' of height h and width w such that $\Delta^{\mathfrak{D}}(S', S' + \tilde{\alpha}_N^{\tau(n)}(u,v)) \geq \epsilon'(n)$, then there is a box S of the same dimensions and with

$$\Delta^{\mathfrak{D}}(S, S + \alpha_N^{\tau(n)}(u,v)) \geq \frac{\epsilon'(n)}{2^{\tau(n)}} - \frac{2}{h} - \frac{2}{w}.$$

Proof. Let $k = [-v^{-1}N]_{2^{\tau(n)}}$. Then

$$\begin{aligned} k\alpha_N^{\tau(n)}(u,v) &\equiv [-v^{-1}N]_{2^{\tau(n)}}(u2^{i+1-\tau(n)} + v[2^{-\tau}]_N) \equiv \\ &\equiv ([-uv^{-1}N]_{2^{\tau(n)}} + c_1 2^{\tau(n)})2^{i+1-\tau(n)} + (-N + c_2 2^{\tau(n)})[2^{-\tau(n)}]_N \equiv \\ &\equiv [-uv^{-1}N]_{2^{\tau(n)}} 2^{i+1-\tau(n)} + c_1 2^{i+1} + c_2 \bmod N, \end{aligned}$$

where $0 \leq c_1 < 2^{\tau(n)}$ and $0 \leq -N + c_2 2^{\tau(n)} \leq 2^{2\tau(n)}$. This implies that

$$k\alpha_N^{\tau(n)}(u,v) - \tilde{\alpha}_N^{\tau(n)}(u,v) = c_1 2^{i+1} + c_2' \bmod N,$$

where $c_2' = c_2 - \lceil \frac{N}{2^{\tau(n)}} \rceil$ and hence $0 \leq c_2' < 2^{\tau(n)}$. We conclude that

$$\#\left(\left(S' + \tilde{\alpha}_N^{\tau(n)}(u,v)\right) \triangledown \left(S' + k\alpha_N^{\tau(n)}(u,v)\right)\right) \leq 2c_1 w + 2c_2' h.$$

Hence

$$\Delta^{\mathfrak{D}}(S', S' + k\alpha_N^{\tau(n)}(u,v)) \geq \epsilon'(n) - \frac{2c_1}{h} - \frac{2c_2'}{w},$$

and the existence of k follows by the triangle inequality. □

Lemma 5.16 allows us to study sequences of the form

$$\{j\tilde{\alpha}_N^{\tau(n)}(u,v)\}_{j\geq 0} = \{j(u'2^{i+1-\tau(n)} + \lceil N/2^{\tau(n)} \rceil)\}_{j \geq 0},$$

where $u' = [-uv^{-1}N]_{2^\tau}$, rather than $\{j\alpha_N^{\tau(n)}(u,v)\}_{j\geq 0}$. The key benefit of this is that the former sequence is strictly increasing with respect to $\pi_Y(\cdot)$. Also, since $u' < 2^{\tau(n)}$ and $2^{i+1} < N/2^{2\tau(n)}$ (from the upper bound on i), we never need to perform any modular reductions modulo N, i.e.

$$[j(u'2^{i+1-\tau(n)} + \lceil N/2^{\tau(n)}\rceil)]_N \equiv j(u'2^{i+1-\tau(n)} + \lceil N/2^{\tau(n)}\rceil), \quad 0 \le j \le 2^{\tau(n)} - 1,$$

and this simplifies the analysis. The central point point of the rest of the proof is to study how the sequence $\{j\tilde{\alpha}_N^{\tau(n)}(u,v)\}_{j\ge 0}$ behaves modulo 2^{i+1}. One key property is whether $\tilde{\alpha}_N^{\tau(n)}(u,v)2^{-(i+1)}$ can be well approximated by a rational number with small denominator. We need some definitions.

Definition 5.17. The number $\zeta \in \mathbb{Q}$ is said to be of (Q,ψ)-*type* if for all integers r, s, $0 < s \le Q$ and $(r,s) = 1$:

$$\left|\zeta - \frac{r}{s}\right| > \frac{1}{s^2\psi}.$$

Definition 5.18. Define $Q(n) \triangleq 2^{10}\epsilon(n)^{-1}$, $\psi(n) \triangleq \frac{\epsilon(n)2^{\tau(n)}}{2^{12}\log^2 Q(n)}$.

We are now ready to state the three main lemmas needed to complete the proof of security of the internal bits of RSA.

Lemma 5.19. Let v be odd. If the rational number $\tilde{\alpha}_N^{\tau(n)}(u,v)/2^{i+1}$ is of $(Q(n),\psi(n))$-type, then there is a box S of width $2^{i+1}\epsilon(n)/8$, height at least $N_3 - 1$, and with $\Delta^{\mathfrak{O}}(S, S + \tilde{\alpha}_N^{\tau(n)}(u,v)) \ge \frac{\epsilon(n)}{2^{\tau(n)+2}}$.

We give the proof in Section 5.3.2 on page 22. The key fact used in the proof is that if $\tilde{\alpha}_N^{\tau(n)}(u,v)$ is of the given type then $\{j\tilde{\alpha}_N^{\tau(n)}(u,v)\}$ is evenly distributed modulo 2^{i+1}.

Finally we need to address the case when we do have very good rational approximations of $\tilde{\alpha}_N^{\tau(n)}(u,v)/2^{i+1}$. The analysis is divided into two cases depending on whether the denominator of this strong rational approximation is odd or even.

Lemma 5.20. Suppose v is odd and that there are relatively prime integers r, s, $0 < s \le Q(n)$ and s even, so that

$$\left|\frac{\tilde{\alpha}_N^{\tau(n)}(u,v)}{2^{i+1}} - \frac{r}{s}\right| \le \frac{1}{s^2\psi(n)}, \tag{5.2}$$

then there is $k \le s$ such that

$$\Delta^{\mathfrak{O}}(S^{(0)} + k\tilde{\alpha}_N^{\tau(n)}(u,v), S^{(0)} + (k+1)\tilde{\alpha}_N^{\tau(n)}(u,v)) \ge \frac{\epsilon(n)}{2s}.$$

The proof is rather similar to the proof for even v (Lemma 5.14) and is given in Section 5.3.3 on page 24.

In the case of a good approximation with an odd denominator we cannot prove that there exists a good box and in fact there are counterexamples showing that there might not be any good boxes. We can prove, however, that if no good box exists, then we can in fact find a related oracle which distinguishes intervals at distance $(N+1)/2$.

Lemma 5.21. Suppose there are integers u, v, r, s, $0 < v \leq 2^{\tau(n)} - 1$, v odd, $|u| \leq 2^{\tau(n)} - 1$, $0 < s \leq Q(n)$, $(r, s) = 1$ and s odd, such that

$$\left| \frac{\tilde{\alpha}_N^{\tau(n)}(u, v)}{2^{i+1}} - \frac{r}{s} \right| \leq \frac{1}{s^2 \psi(n)}, \tag{5.3}$$

and for all boxes S of height at least $sN_1 2^{-\tau(n)}$ and width at least $2^{i+1}\epsilon(n)/(30s)$, we have that $\Delta^{\mathfrak{O}}(S, S + \tilde{\alpha}_N^{\tau(n)}(u, v)) \leq \epsilon(n) 2^{-(\tau(n)+3)}$. Then, using $\mathfrak{O}$, we can in random polynomial time construct an oracle $\mathfrak{O}'$ and find an interval J of length at least $N\epsilon(n)/32$ such that $\Delta^{\mathfrak{O}'}(J, J + (N+1)/2) \geq \epsilon(n)/8$.

The proof is given in Section 5.3.4 on page 25.

We can now add up together the pieces to establish security of all except the most significant bits.

Theorem 5.22. For $i \leq n - 3\tau(n) - \log \epsilon(n)^{-1} - 7$, the ith bit in an RSA encrypted message is secure, unless RSA can be broken in random polynomial time.

Proof. If the hypothesis of Lemma 5.21 is true we can use the constructed $\mathfrak{O}'$ together with Lemma 5.4.

If the hypothesis of Lemma 5.21 is false then Lemma 5.14, Lemma 5.16, Lemma 5.19, and Lemma 5.20 establishes the existence of all boxes needed to apply Lemma 5.12. □

Section 6 considers the remaining bits, $i > n - 3\tau(n) - \log \epsilon(n)^{-1} - 7$.

As promised, we now turn to the postponed proofs. We start with the proof of Lemma 5.14 and remember that it deals with multiples of the original $\alpha_N^{\tau(n)}(u, v)$ and not $\tilde{\alpha}_N^{\tau(n)}(u, v)$ which only is relevant for odd v.

5.3.1 Proof of Lemma 5.14; even v

Setting $v = 2v'$ we have

$$2^{\tau(n)-1} \alpha_N^{\tau(n)}(u, v) \equiv u2^i + v' \mod N.$$

Since u is odd, this implies that

$$\lambda((S^{(0)} + 2^{\tau(n)-1}\alpha_N^{\tau(n)}(u, v)) \nabla S^{(1)}) \leq 2^{\tau(n)} \frac{2^i}{N} + 2^{\tau(n)-i} \leq \epsilon(n)/3.$$

The two error terms comes from $u2^i$ causing a modular reduction modulo N and v' causing a shift modulo 2^{i+1} respectively. The last inequality is due to the definition of $\tau(n)$ and the assumption made on i.

By definition

$$\Delta^{\mathfrak{O}}(S^{(0)}, S^{(1)}) \geq \epsilon(n) - \beta_i(N),$$

where $\beta_i(N)$ is the bias of the ith bit. Since the bias is bounded by $\epsilon(n)/6$ for the range of i we are considering we conclude that

$$\Delta^{\mathfrak{D}}\left(\left(S^{(0)}+2^{\tau(n)-1}\alpha_N^{\tau(n)}(u,v)\right),S^{(0)}\right)\geq\epsilon(n)/2.$$

The existence of the k in the lemma now follows by the triangle inequality.

5.3.2 Proof of Lemma 5.19; $\tilde{\alpha}_N^{\tau(n)}(u,v)2^{-(i+1)}$ of $(Q(n),\psi(n))$-type

The famous Weyl equidistribution theorem states that if ζ is irrational, the fractional parts of the sequence $\{j\zeta\}_{j=0}^{K-1}$ are uniformly distributed in $[0,1]$ in the sense that as $K\rightarrow\infty$, each $[a,b]\subset[0,1]$, gets about the expected number of points from the sequence, i.e. a $b-a$ fraction. The rate of convergence to the uniform distribution depends on the extent to which ζ is approximable by rationals. The assumption on $\tilde{\alpha}_N^{\tau(n)}(u,v)$ implies, through a quantitative version of the Weyl theorem, that $\{j\alpha_N^{\tau(n)}(u,v)\}_{j=0}^{2^{\tau(n)}-1}$ is nicely distributed modulo 2^{i+1} and this is the key fact that we use in this section, see Theorem 5.25. Let us start by defining a set of boxes.

Definition 5.23. Let $w(n)=2^{i+1}\epsilon(n)/8$, $m(n)=\left\lfloor 2^{i+1}/w(n)\right\rfloor$, $h(n)=\pi_Y(\tilde{\alpha}_N^{\tau(n)}(u,v))$ and let $S_{0,0}$ be the box $[0..w(n)-1]\times[0..h(n)-2]$. Define

$$S_{j,k}=S_{0,0}+jw(n)+k\tilde{\alpha}_N^{\tau(n)}(u,v)$$

for $0\leq j\leq m(n)-1$ and $0\leq k\leq 2^{\tau(n)}-2$. A box is *split* if it intersects both $S^{(0)}$ and $S^{(1)}$. Define the *orbit* o_j by

$$o_j=\bigcup_k S_{j,k}$$

where the union is only taken over boxes that are not split.

Figure 4 below describes the boxes $S_{j,k}$ in a picture.

We establish the basic properties of our set of boxes.

Lemma 5.24. The boxes $\{S_{j,k}\}$ are pairwise disjoint and cover $\Pi(N,i)$ except for at most a $\epsilon(n)/2$-fraction. The total measure of the split boxes is at most $\epsilon(n)/8$.

Proof. First of all, notice that since

$$w(n)-1+(h(n)-2)2^{i+1}+(\left\lfloor 2^{i+1}/w(n)\right\rfloor-1)w(n)+(2^{\tau(n)}-2)\tilde{\alpha}_N^{\tau(n)}(u,v)\leq$$
$$(h(n)-1)2^{i+1}+(2^{\tau(n)}-2)(2^i+\frac{N}{2^{\tau(n)}})<N$$

we need not perform any modular reductions when studying the boxes $S_{j,k}$. The boxes are disjoint since boxes with different k-values have disjoint projections on

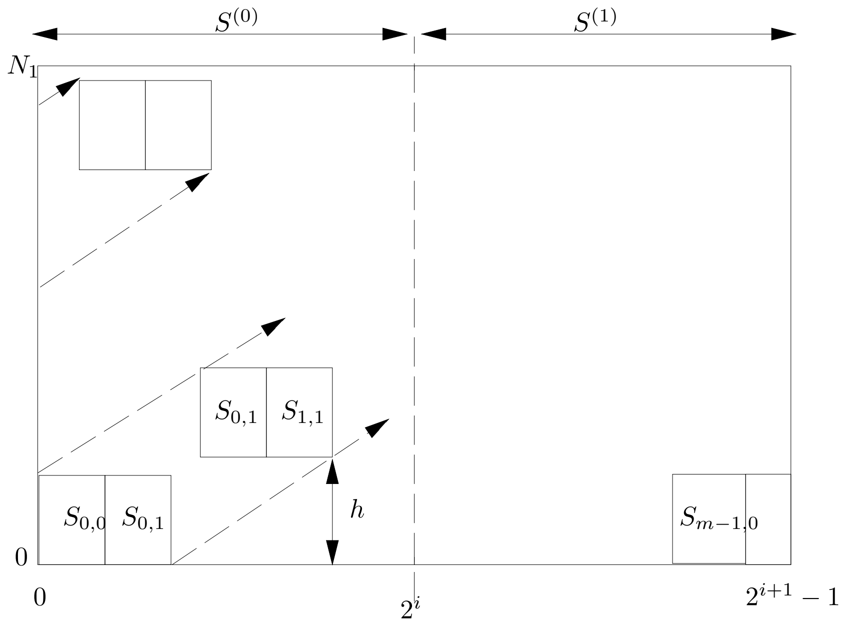

Figure 4: The basic boxes.

the Y-axis and boxes with the same k-value and different j-values have disjoint projections on the I-axis. The total size of the boxes is

$$(h(n)-1)w(n)m(n)(2^{\tau(n)}-1) \geq (2^{i+1}-w(n))(1-2^{1-\tau(n)})N_1 \geq (1-\epsilon(n)/4)N$$

and thus they cover all but an $\epsilon(n)/4$ fraction of the plane. Finally note that for each k only one $S_{j,k}$ is split and thus we have at most $2^{\tau(n)}$ split boxes and the total size of these split boxes is bounded by $2^{\tau(n)}(h(n)-1)w(n) \leq \epsilon(n)N/8$. □

As another preliminary consider the below theorem, the proof of which we postpone to the appendix.

Theorem 5.25. Let $0 \leq v \leq 2^{\tau(n)}-1$, v odd, $|u| \leq 2^{\tau(n)}-1$. If $\tilde{\alpha}_N^{\tau(n)}(u,v)/2^{i+1} \in \mathbb{Q}$ is of $(Q(n),\psi(n))$-type, then for all $0 \leq a < b < 2^{i+1}$,

$$\left|\Pr_j[a \leq j\tilde{\alpha}_N^{\tau(n)}(u,v) \leq b] - \frac{b-a}{2^{i+1}}\right| \leq 14\left(\frac{1}{Q(n)} + \frac{4\psi(n)\log^2 Q(n)}{2^{\tau(n)}}\right),$$

the probability taken over j, chosen uniformly at random in $\{0,1,\ldots,2^{\tau(n)}-2\}$.

Let us now turn to the proof of Lemma 5.19. In view of Lemma 5.24, $\mathfrak{O}$ must have advantage $\epsilon(n)/2$ of determining the i'th bit on o_{j_0} for some j_0. Each individual box that is part of o_{j_0} is not split and hence it is either contained completely in $S^{(0)}$ or completely in $S^{(1)}$. Define $o_{j_0,k} = o_j \cap S^{(k)}$ and assume that $o_{j_0,k}$ contains n_k boxes. Since being contained in $S^{(0)}$ is equivalent to the lower left hand corner being in an interval of length $2^i - w(n)$ modulo 2^{i+1}, and the same is true for being contained in $S^{(1)}$, two applications of Theorem 5.25 yield

$$|n_1 - n_0| \leq 28(2^{\tau(n)} - 1)\left(\frac{1}{Q(n)} + \frac{4\psi(n)\log^2 Q(n)}{2^{\tau(n)}}\right) \leq 2^{\tau(n)}\epsilon(n)/16 \qquad (5.4)$$

and an additional application (using very blunt estimates) of the same theorem yields

$$n_1 + n_0 \geq 2^{\tau(n)}/2 \qquad (5.5)$$

Assume for concreteness that $n_1 \geq n_0$. Now pair each box in $o_{j_0,0}$ in some arbitrary way with a unique box in $o_{j_0,1}$. By (5.4) and (5.5), at most a fraction $\epsilon(n)/8$ of the boxes remain single. Thus by the assumption on the oracle there must be an ℓ_k, $k = 0, 1$ such that $S_{j_0,\ell_k} \in o_{j_0,k}$ and such that $\mathfrak{O}$ has advantage at least $\epsilon(n)/4$ over $S_{j_0,\ell_0} \cup S_{j_0,\ell_1}$. Now, since $S_{j_0,\ell_k} \subset S^{(k)}$ we can conclude that

$$\Delta^{\mathfrak{O}}(S_{j_0,\ell_0}, S_{j_0,\ell_1}) \geq \epsilon(n)/4$$

. The lemma now follows by the triangle inequality.

5.3.3 Proof of Lemma 5.20; even denominator s

Set $s = 2s'$ and consider $s'\tilde{\alpha}_N^{\tau(n)}(u,v)$. By the assumption on $\tilde{\alpha}_N^{\tau(n)}(u,v)$ and using that r is odd we have

$$|\pi_I(s'\tilde{\alpha}_N^{\tau(n)}(u,v)) - 2^i| \leq \frac{2^{i+1}}{s\psi(n)}.$$

Furthermore $|s'\tilde{\alpha}_N^{\tau(n)}(u,v)| \leq Q(n)N2^{-\tau(n)}$. This implies that

$$\lambda\left(\left(S^{(0)} + s'\alpha_N^{\tau(n)}(u,v)\right)\nabla S^{(1)}\right) \leq \frac{2Q(n)}{2^{\tau(n)}} + \frac{2}{s\psi(n)}.$$

By the choice of $Q(n)$ and $\tau(n)$ this latter quantity is bounded from above by $\epsilon(n)/3$. Now,

$$\Delta^{\mathfrak{O}}(S^{(0)}, S^{(1)}) \geq \epsilon(n) - \beta_i(N)$$

where $\beta_i(N)$ is the bias of the ith bit. Since this is, by the assumption on i, small compared to $\epsilon(n)$ we conclude that

$$\Delta^{\mathfrak{O}}\left(\left(S^{(0)} + s'\alpha_N^{\tau(n)}(u,v)\right)\nabla S^{(0)}\right) \geq \epsilon(n)/2.$$

The existence of k now follows by the triangle inequality.

5.3.4 Proof of Lemma 5.21; odd denominator s

To see how the proof will go, we remind the reader of the work by Ben-Or et al. in [3]. Ben-Or et al. showed that if $\mathfrak{O}$ is an $\epsilon(n)$-oracle for the ith bit in $E_N^{-1}(x)$, and we, utilizing the multiplicative properties of RSA, define a new oracle, $\mathfrak{O}_2$, by

$$\mathfrak{O}_2(E_N(x)) = \mathfrak{O}(E_N([N_1^{-1}x]_N)), \tag{5.6}$$

then $\mathfrak{O}_2(E_N(x))$ distinguishes between some sets $J, J+(N+1)/2$, increasing the error probability of $\mathfrak{O}$ by a quantity depending on $[N]_{2^{i+1}}$ and this quantity in turn is $\frac{1}{4}$ in the worst case (a tight bound). Using the improved sampling techniques from [1], a $\frac{1}{2}$-security result for the internal RSA bits follows. The reason that this works is that the mapping $z \mapsto [N_1 z]_N$ maps intervals at distance 2^i to intervals "almost" at distance $(N+1)/2$. This "almost" depends on $[N]_{2^{i+1}}$ and gives rise to the additional error term.

The assumptions of Lemma 5.21 enables us to find another transformation (similar to (5.6)) of the original oracle that maps certain sets at distance 2^i to sets also almost at distance $(N+1)/2$ and where the oracle has a significant advantage. We start by a preliminary lemma.

Lemma 5.26. If there are integers u, v, r, s, $0 < v \le 2^{\tau(n)} - 1$, v odd, $|u| \le 2^{\tau(n)} - 1$, $0 < s \le Q(n)$, $(r,s) = 1$ and s odd, such that $\left|\frac{\tilde{\alpha}_N^{\tau(n)}(u,v)}{2^{i+1}} - \frac{r}{s}\right| \le \frac{1}{s^2\psi(n)}$, then for $u' = [-uv^{-1}N]_{2^{\tau(n)}}$ there is $r' \in \mathbb{Z}$, $r' \le 2Q(n)$ so that for all sufficiently large n,

$$\left|s(u'+N_2) - r'2^{\tau(n)}\right| \le ns\epsilon(n)^{-1}.$$

Proof. Set $r' = r - sN_3$. Unfolding the definition of $\tilde{\alpha}_N^{\tau(n)}(u,v)$, for some $\delta < 2^{\tau(n)}$ we have

$$\begin{aligned}
\left|\frac{\tilde{\alpha}_N^{\tau(n)}(u,v)}{2^{i+1}} - \frac{r}{s}\right| &= \left|\frac{u'2^{i+1} + N_3 2^{i+1+\tau(n)} + N_2 2^{i+1} + N_0 + \delta}{2^{i+1+\tau(n)}} - \frac{r}{s}\right| \\
&= \left|N_3 + \frac{u'2^{i+1} + N_2 2^{i+1} + N_0 + \delta}{2^{i+1+\tau(n)}} - \frac{r'}{s} - N_3\right| \\
&= \left|\frac{(u'+N_2)2^{i+1} + N_0 + \delta}{2^{i+1+\tau(n)}} - \frac{r'}{s}\right|.
\end{aligned}$$

Multiplying by $2^{\tau(n)}s$ and using the assumption we get:

$$\left|s(u'+N_2) + \frac{s(N_0+\delta)}{2^{i+1}} - 2^{\tau(n)}r'\right| \quad \le \quad 2^{\tau(n)}\frac{1}{s\psi(n)}.$$

But $N_0 + \delta \le 2^{i+1}$, so

$$\left|s(u'+N_2) - 2^{\tau(n)}r'\right| \le 2^{\tau(n)}\frac{1}{s\psi(n)} + s.$$

Using $s \leq Q(n)$, $u' < 2^{\tau(n)}$, $N_2 < 2^{\tau(n)}$ and substituting the definition of $Q(n), \psi(n)$, and $\tau(n)$ now establishes the results. □

The integer $s(u' + N_2) - 2^{\tau(n)} r'$ plays a special role in our argument and we introduce the symbol κ for it.

Definition 5.27. Define the integer

$$\kappa \triangleq s(u' + N_2) - 2^{\tau(n)} r'.$$

In the remainder of this section we now concentrate on r', s, u', κ as above. We can at this point write down the oracle that distinguishes between some J and $J + (N+1)/2$.

Definition 5.28. Define $\varphi : \mathbb{Z}_N \to \mathbb{Z}_N$ by

$$\varphi(z) \triangleq [(sN_1 - \kappa)z]_N.$$

For $S \subset \mathbb{Z}_N$, $\varphi(S)$ is defined in the natural way; $\{\varphi(z) \mid z \in S\}$.

We now define the oracle

$$\mathfrak{O}'(E_N(x)) \triangleq \mathfrak{O}(E_N(\varphi^{-1}(x))).$$

We see that when $s = 1, \kappa = 0$, we get precisely the same oracle construction as in [3].

It may be the case that φ^{-1} does not exist, i.e. that $sN_1 - \kappa$ does not have a multiplicative inverse. If this happens then we have factored[3] N and we can invert RSA. Hence we may assume that φ^{-1} exists.

We now study the behavior of $\mathfrak{O}$ on certain boxes.

Definition 5.29. Let

$$w'(n) \triangleq \left\lfloor 2^{i+1}\left(\frac{1}{2s} - \frac{1}{s\psi(n)}\right) \right\rfloor$$

and $w(n) \triangleq \lfloor w'(n)\epsilon(n)/10 \rfloor$. Define the base box

$$S_{0,0} \triangleq \{0, \dots, w(n) - 1\} \times \{0, \dots, \pi_Y(s\tilde{\alpha}_N^{\tau(n)}(u,v)) - 1\}$$

and then translated boxes

$$S_{j,k} \triangleq S_{0,0} + k\tilde{\alpha}_N^{\tau(n)}(u,v) + jw(n), \qquad 0 < k < 2^{\tau(n)} - s, \quad 0 \leq j < \lfloor w'(n)/w(n) \rfloor.$$

Also, define the orbit

$$o_j \triangleq \bigcup_k S_{j,k}.$$

For each $S_{j,k}, o_j$ we define $S'_{j,k} \triangleq S_{j,k} + 2^i$, $o'_j \triangleq o_j + 2^i$.

As before, we call a box S *split* if both $S \cap S^{(0)}$, and $S \cap S^{(1)}$ are non-empty.

[3] Note that $sN_1 - \kappa$ is much smaller than N and it is much larger than 0.

The proof will now proceed as follows. By assumption, $\mathfrak{O}$ behaves almost the same on all boxes within any fixed orbit. We will shortly see (in Lemmas 5.31 and 5.32), that under the mapping $\varphi(\cdot)$, o_j gets mapped into what is (almost) an interval J_j, and that o'_j (almost) maps to $J_j + (N+1)/2$. We prove that if $\mathfrak{O}$ has a significant advantage in guessing the ith bit on $S_{j,k} \cup S'_{j,k}$ for some k, then $\mathfrak{O}'$ distinguishes J_j and $J_j + (N+1)/2$. We establish that the boxes cover most of the plane and hence there must be such a j and this completes the argument. We start by investigating how well the boxes $S_{j,k}$ and $S'_{j,k}$ cover the $\Pi(N,i)$-plane.

Lemma 5.30. The collection of boxes given by all $S_{j,k}$, and $S'_{j,k}$ for $0 \leq j < \lfloor w'(n)/w(n) \rfloor$ and $0 \leq k < 2^{\tau(n)} - s$ are disjoint and cover the plane except for a fraction at most $\epsilon(n)/4$. The total measure of all split boxes is at most $\epsilon(n)/10$.

Proof. First we claim that no modular reductions are needed in the definition of the boxes. This follows since the maximal value of any element in any of the boxes is bounded by

$$(2^{\tau(n)} - (s+1))\tilde{\alpha}_N^{\tau(n)}(u,v) + s\tilde{\alpha}_N^{\tau(n)}(u,v) + 2^{i+1} < N.$$

Next note that $S_{j,k}$ are disjoint for different j and a fixed value of k and thus we can study the "superboxes"

$$B_k \triangleq \bigcup_j S_{j,k}$$

together with their similarly defined counterparts B'_k. The width of such a superbox is bounded by $w'(n)$. By symmetry and translation we need only prove that for any k, neither B_k nor B'_k intersect B_0. Since B'_0 clearly does not intersect B_0, by studying Y-coordinates it follows that we need only consider $0 < k < s$. Now the lower left corner of B_k and B'_k has I-coordinates $\pi_I(k\tilde{\alpha}_N^{\tau(n)}(u,v))$ and $\pi_I(k\tilde{\alpha}_N^{\tau(n)}(u,v)) + 2^i$, respectively. For a box to intersect with B_0 this coordinate should be at least $2^{i+1} - w'(n)$. By (5.3) on page 21, setting $\ell = kr$ modulo s, we see that $k\tilde{\alpha}_N^{\tau(n)}(u,v)$ modulo 2^{i+1} is within distance at most $2^{i+1}(s\psi(n))^{-1}$ of $\ell 2^{i+1}/s$. Since ℓ is not 0, this number attains its maximal value when $\ell = s-1$. To have an intersection of B_k with B_0 we would need

$$\frac{s-1}{s}2^{i+1} + 2^{i+1}\frac{1}{s\psi(n)} \geq 2^{i+1} - w'(n)$$

but

$$2^{i+1}\frac{1}{s\psi(n)} + w'(n) < \frac{2^{i+1}}{2s} \tag{5.7}$$

and thus we can have no intersection. The largest possible value of the lower left corner of B'_k is obtained when when $\ell = (s-1)/2$ and in this case the condition of intersection is

$$\frac{2s-1}{2s}2^{i+1} + 2^{i+1}\frac{1}{s\psi(n)} \geq 2^{i+1} - w'(n),$$

which again is false by (5.7). Thus the boxes are disjoint.

The size of each $S_{j,k}$ is $w(n)\pi_Y(s\tilde{\alpha}_N^{\tau(n)}(u,v))$ and the number of boxes of each of the two types is at least $(2^{\tau(n)}-s)(w'(n)/w(n)-1)$. Thus the total size of all the boxes is

$$\begin{aligned} 2(2^{\tau(n)}-s)(w(n)'/w(n)-1)w(n)\pi_Y(s\tilde{\alpha}_N^{\tau(n)}(u,v)) &\geq \\ \left\lfloor \frac{N}{2^{\tau(n)+i+1}} \right\rfloor (2^{\tau(n)}-s)2s(w'(n)-w(n)) &\geq \\ \frac{N}{2^{i+1}}(1-\frac{2s}{2^{\tau(n)}})2^{i+1}(1-\frac{2}{\psi(n)})(1-\epsilon(n)/10) &\geq N(1-\epsilon(n)/4). \end{aligned}$$

Finally let us study the size of the split boxes. Any split box intersects the middle vertical line (i.e. $\pi_I(x)=2^i$) for $\pi_Y(s\tilde{\alpha}_N^{\tau(n)}(u,v))$ levels. Since there are only $N2^{-(i+1)}$ levels we have at most $2^{\tau(n)}/s$ split boxes. Their total measure is at most $w(n)\leq\epsilon(n)/10$. The proof is complete. □

We proceed by investigating how φ acts on the $\Pi(N,i)$-plane. Of particular interest is what happens to the number $\tilde{\alpha}_N^{\tau(n)}(u,v)$ and what happens with values that differ in the ith bit position. We start by estimating $\varphi(\tilde{\alpha}_N^{\tau(n)}(u,v))$.

Lemma 5.31.

$$|\varphi(\tilde{\alpha}_N^{\tau(n)}(u,v))| \leq 2^{12}ns\epsilon(n)^{-1}\max(2^{i+1},N/2^{i+1}).$$

Proof. We need to estimate $(sN_1-\kappa)\tilde{\alpha}_N^{\tau(n)}(u,v)$. Let us for the moment ignore the term $\kappa\tilde{\alpha}_N^{\tau(n)}(u,v)$ and concentrate on $sN_1\tilde{\alpha}_N^{\tau(n)}(u,v)$. Since N_1 is close to $N2^{-(i+1)}$ it is useful to write $s\tilde{\alpha}_N^{\tau(n)}(u,v)$ on the form $a2^{i+1}+b$ for integers a and b. Introducing $\delta<2^{\tau(n)}$, so that $N+\delta$ is divisible by $2^{\tau(n)}$, with $u'=[-uv^{-1}N]_{2^{\tau(n)}}$, we have

$$\begin{aligned} s\tilde{\alpha}_N^{\tau(n)}(u,v) &= s\frac{u'2^{i+1}+N+\delta}{2^{\tau(n)}} = s\frac{(u'+N_3 2^{\tau(n)}+N_2)2^{i+1}+N_0+\delta}{2^{\tau(n)}} \\ &= sN_3 2^{i+1} + s\frac{(u'+N_2)2^{i+1}+N_0+\delta}{2^{\tau(n)}} \\ &= sN_3 2^{i+1} + \frac{(\kappa+2^{\tau(n)}r')2^{i+1}+s(N_0+\delta)}{2^{\tau(n)}} \\ &= (sN_3+r')2^{i+1}+\kappa 2^{i+1-\tau(n)}+\frac{s(N_0+\delta)}{2^{\tau(n)}} \qquad (5.8) \end{aligned}$$

Now $N_1 2^{i+1}\equiv -N_0$ modulo N and hence using (5.8)

$$sN_1\tilde{\alpha}_N^{\tau(n)}(u,v)\equiv -N_0(sN_3+r')+N_1\kappa 2^{i+1-\tau(n)}+\frac{sN_1(N_0+\delta)}{2^{\tau(n)}} \bmod N.$$

Now $|r'N_0|\leq 2^{i+1}2Q(n)\leq 2^{11}\epsilon(n)^{-1}2^{i+1}$ and $|s\delta N_1 2^{-\tau(n)}|\leq sN2^{-i}$. Furthermore

$$sN_1N_0 2^{-\tau(n)}-sN_0N_3=sN_0N_2 2^{-\tau(n)}$$

and this is of absolute value at most $s2^i$. Remembering the omitted term $\kappa\tilde{\alpha}_N^{\tau(n)}(u,v)$ we have

$$N_1\kappa 2^{i+1-\tau(n)} - \kappa\tilde{\alpha}_N^{\tau(n)}(u,v) = \kappa(N_0 + \delta + u'2^{i+1})2^{-\tau(n)}$$

which is of absolute value at most $\kappa 2^{i+2}$. Collecting the error terms, the lemma follows. □

It may seem that the error term $\sim \max(2^{i+1}, N/2^{i+1})$ is very large. However, since the plan is to find intervals $J, J + (N+1)/2$ where the oracle behaves differently, the error term should be compared to N and for the range of i currently under consideration our error is small compared to N.

Lemma 5.32. For sufficiently large n,

$$\left|\varphi(2^i) - \frac{N+1}{2}\right| \leq 2sn\epsilon(n)^{-1}2^{i+1}$$

and

$$\text{abs}_N\left(\varphi(2^{i+1})\right) \leq 4sn\epsilon(n)^{-1}2^{i+1}$$

Proof. To study $\varphi(2^i) = [(sN_1 - \kappa)2^i]_N$ we first note that $|\kappa 2^i| \leq ns\epsilon(n)^{-1}2^i$ and this will be part of the error term. Writing $s = 2s' + 1$ for an integer s' we see that

$$sN_1 2^i = s'N_1 2^{i+1} + N_1 2^i.$$

Now $N_1 2^{i+1} \equiv -N_0$ modulo N and $|s'N_0| \leq s2^{i+1}$. Noting that $|N_1 2^i - (N + 1)/2| \leq 2^i$, we establish the first part of the lemma by collecting the error terms. The second part of the lemma is follows immediately from the first. □

The first part of the Lemma says that values that differ in their ith bit gets mapped to values essentially $(N+1)/2$ apart.

We now study how orbits, o_j, o'_j can be mapped into intervals.

Lemma 5.33. There is an interval J_j of length at least $N\epsilon(n)/32$ such that

$$\#\left(J_j \nabla \varphi(o_j)\right) \leq \epsilon(n)w(n)sN_1/16$$

and

$$\#\left(\left(J_j + \frac{N+1}{2}\right)\nabla\varphi(o'_j)\right) \leq \epsilon(n)w(n)sN_1/16.$$

Proof. Define J_j as $[jsN_1w(n), \ldots (j+1)sN_1w(n)-1]$. The length of this interval is

$$\#J_j = sN_1w(n) \geq w'(n)\epsilon(n)sN_1/11 \geq \epsilon(n)2^{i+1}N_1/23 \geq \epsilon(n)N/32.$$

The orbit o_j contains $(2^{\tau(n)} - s)\pi_Y(s\tilde{\alpha}_N^{\tau(n)}(u,v))w(n)$ points. As a first part to establish the claim we prove that the sizes of the two sets (i.e. J_j and $\varphi(o_j)$) are about equal. To see this, note that $\pi_Y(s\tilde{\alpha}_N^{\tau(n)}(u,v))$ is within 1 of $sN2^{-(i+1+\tau(n))}$ which in its turn is within 1 of $sN_1 2^{-\tau(n)}$. Thus the total number of points in o_j is of the form $(1+\delta(n))sN_1 w(n)$ where

$$|\delta(n)| \leq (s+2)2^{-\tau(n)} \leq \epsilon(n)/64.$$

To establish the first part of the lemma we thus just need to prove that at most a fraction $\epsilon(n)/32$ of the points of o_j are mapped outside J_j by φ.

Let us first consider the bottom level of $S_{j,0}$. If it was not for the presence of κ in the definition of φ this bottom level would have been mapped evenly to the entire J_j. However the presence of κ only displaces elements of this bottom level at most $\kappa 2^i$ which is bounded by $|J_j|\epsilon(n)/128$.

Let us next consider the bottom levels of $S_{j,k}$. By Lemma 5.31 these are only shifted a distance at most

$$2^{\tau(n)} 2^{12} ns\epsilon(n)^{-1} \max(2^{i+1}, N/2^{i+1})$$

which is again bounded by $|J_j|\epsilon(n)/128$.

Finally let us consider the non-bottom levels. By Lemma 5.32 starting points of adjacent levels get mapped to points only $4sn\epsilon(n)^{-1}2^{i+1}$ apart. Since we have $sN_1 2^{-\tau(n)}$ levels in one box the top layer has been shifted a distance $4s^2 n\epsilon(n)^{-1}2^{-\tau(n)}N$. This is, by the choice of $\tau(n)$, bounded by $|J_j|\epsilon(n)/128$. Adding the error terms we get the first part of the lemma.

To study the behavior of o'_j we need only add the extra error term $2sn\epsilon^{-1}(n)2^{i+1}$, as given by Lemma 5.32 coming from that fact that 2^i is not mapped exactly to $(N+1)/2$. This small extra term does not disturb the calculations. □

We get immediately.

Corollary 5.34. If there is a j such that $\Delta^{\mathfrak{O}}(o'_j, o_j) \geq \epsilon(n)/4$ then there is an interval J_j, of length at least $\epsilon(n)N/32$ for which the oracle $\mathfrak{O}'$ has

$$\Delta^{\mathfrak{O}'}(J_j, J_j + (N+1)/2) \geq \frac{\epsilon(n)}{8}.$$

The last piece in the proof of Lemma 5.21 is given by the following lemma.

Lemma 5.35. If $\mathfrak{O}$ has advantage $\epsilon(n)$ in deciding the ith bit then for some j we have $\Delta^{\mathfrak{O}}(o'_j, o_j) \geq \epsilon(n)/4$.

Proof. When considering the oracle only on the part of $\mathbb{Z}_N$ covered by nonsplit boxes of the form $S_{j,k}$ or $S'_{j,k}$ the oracle must, by Lemma 5.30, still have advantage $\epsilon(n)/2$. Since $\mathfrak{O}$ must achieve its average somewhere there must be a pair on nonsplit boxes $(S_{j,k}, S'_{j,k})$ such that $\mathfrak{O}$ has advantage at least $\epsilon(n)/2$ in predicting the ith bit on $S_{j,k} \cup S'_{j,k}$. Since the ith bit is constant on both $S_{j,k}$ and $S'_{j,k}$ and different on these two sets we can conclude that $\Delta^{\mathfrak{O}}(S_{j,k}, S'_{j,k}) \geq \epsilon(n)/2$. Now by assumption on $\mathfrak{O}$ for any l we have

$$\Delta^{\mathfrak{O}}(S_{j,l}, S_{j,k}) \leq |k-l| 2^{-(\tau(n)+3)}\epsilon(n) \leq \epsilon(n)/8.$$

This implies that $\Delta^{\mathfrak{O}}(o_j, S_{j,k}) \leq \epsilon(n)/8$ and by a similar reasoning $\Delta^{\mathfrak{O}}(o'_j, S'_{j,k}) \leq \epsilon(n)/8$. By the triangle inequality we conclude that $\Delta^{\mathfrak{O}}(o'_j, o_j) \geq \epsilon(n)/4$. □

We can now draw the final conclusion, proving Lemma 5.21. By Lemma 5.35 we get a pair of orbits on which $\mathfrak{O}$ behaves differently. By Corollary 5.34 this gives the desired pairs of intervals.

6 Security of Leftmost RSA Bits

We now study the $O(\log n)$ most significant bits. A new concern for the most significant bits is that due to a possibly large bias of the ith bit, the oracle's advantage may be severely shifted, favoring values having the ith bit equal to 0. Furthermore, one may argue that if the probability that the ith bit equals 0 is non-negligibly larger than $1/2$, there is a trivial prediction algorithm, one that always predicts '0'.

It has been shown that the definition of $\epsilon(n)$-security used up until now does not generalize in the natural way to functions that are a priori known to be biased. Schrift and Shamir [27] gave the correct definition of "unpredictability" for biased functions. To make the situation interesting we assume that a predicate is non-constant which means that it has a non-neglible probability of outputting both values. The are now several equivalent ways to define unpredictable and we here give the definition that is easiest to apply in the current situation. For other, equivalent, definitions we refer to [27].

Definition 6.1. Let p be a non-constant predicate. An oracle $\mathfrak{O}$ predicts p with advantage $\epsilon(n)$ if

$$|\Pr[\mathfrak{O}(E_N(x)) = 1 \mid p(x) = 1] - \Pr[\mathfrak{O}(E_N(x)) = 1 \mid p(x) = 0]| \geq \epsilon(n). \quad (6.1)$$

A predicate is $\epsilon(n)$*-secure* if no pptm oracle exists with advantage $\epsilon(n)$ and it is *unpredictable* if it is $\epsilon(n)$-secure for all non-negligible $\epsilon(n)$.

Before continuing with the proof, we note that all that appears to be known about the security of the most significant bits in RSA is that certain predicates such as $\text{half}_N(x) = 1$ if $x \geq (N+1)/2$, 0 otherwise, are secure (see [7] for instance). The proof is easy, since as we have seen, this predicate is reducible to/from an lsb-computation: $\text{half}_N(x) = \text{lsb}([2x]_N)$ and $\text{lsb}(x) = \text{half}_N([2^{-1}x]_N)$. This predicate is to some extent, depending on N, related to the most significant bit of x.

6.1 Proof Outline

For RSA it is known (see [1]), that the $t(n) \in O(\log n)$ least significant bits of x are simultaneously secure, i.e. given $E_N(x)$, they are polynomially indistinguishable from random bits. Clearly, this implies that it is infeasible to *predict* these bits with a non-negligible advantage over the trivial $2^{-t(n)}$. The plan is therefore to prove that an $\epsilon(n)$-oracle for $\text{bit}_i(x)$, $i = n - O(\log n)$, can be converted into an algorithm $\mathfrak{O}'$ that for some $t(n) \in O(\log n)$ predicts $\text{B}_0^{t(n)-1}(x)$ with probability $2^{-t(n)} + \epsilon'(n)$, where $\epsilon'(\cdot)$ is non-negligible. This will then give a contradiction to the result in [1].

For the moment, let us assume that the bias of the ith bit is small. Ask the oracle $\mathfrak{O}$ about $\mathrm{bit}_i(E_N^{-1}([2^{-t}x]_N))$ where $t = n - i + t_0$ and where $t_0 \in \Theta(\log n)$ (so that $t \in O(\log n)$). Again we note that

$$[2^{-t}x]_N = \frac{x - \mathrm{B}_0^{t-1}(x)}{2^t} + \mathrm{B}_0^{t-1}(x)[2^{-t}]_N.$$

The term $\frac{x-\mathrm{B}_0^{t-1}(x)}{2^t}$ is small,

$$\frac{x - \mathrm{B}_0^{t-1}(x)}{2^t} \leq \frac{N}{2^t} \leq 2^{i-t_0},$$

so except with probability $\sim 2^{-t_0}$, we have $\mathrm{bit}_i(\mathrm{B}_0^{t-1}(x)[2^{-t}]_N) = \mathrm{bit}_i([2^{-t}x]_N)$. This means that although there are a priori 2^t possibilities for $\mathrm{B}_0^{t-1}(x)$, if the oracle is correct on the ith bit of $[2^{-t}x]_N$, we can narrow it down to roughly 2^{t-1} as only half of the $\mathrm{B}_0^{t-1}(x)$-values would have given this particular value for the ith bit. We now have an algorithm that computes $\mathrm{B}_0^{t-1}(x)$ with probability $2^{-(t-1)}$, which is twice the success rate of any trivial guessing-strategy. We now turn to a formal argument taking also the bias into account.

We analyze the success probability of the following algorithm. $\mathfrak{O}$ is the oracle that is assumed to predict the ith bit of x.

Algorithm 6.2.

```
Input: $E_N(x)$, $\|N\| = n$
Output: $\mathrm{B}_0^{t(n)-1}(x)$, for some $t(n) = n - i + t_0(n) \in O(\log n)$
(1)  $b \leftarrow \mathfrak{O}(E_N([2^{-t(n)}x]_N))$            /* $\mathrm{bit}_i([2^{-t(n)}x]_N)$ */
(2)  $\mathfrak{J} \leftarrow \{j \mid 0 \leq j < 2^{t(n)} \wedge \exists z, 0 \leq z \leq 2^{i-t_0(n)}$ s.t. $\mathrm{bit}_i([j2^{-t(n)} + z]_N) = b\}$
(3)  pick $j \in_{\mathcal{U}} \mathfrak{J}$
(4)  return $j$
```

Notice that for $t(n) \in O(\log n)$, $t_0(n) \geq 1$, the algorithm is polynomial time: For each j, $0 \leq j < 2^{t(n)}$, we only need to consider $z = 0$ and $z = 2^{i-t_0(n)}$ to determine the set $\mathfrak{J}$.

Lemma 6.3. Suppose that $\mathfrak{O}$ satisfies (6.1) of Definition 6.1 and that the bias is upper bounded by $\beta_i(N) \leq 1 - \delta(n)$ where $\delta(n)$ is non-negligible. Then, for $t(n) = n - i + t_0(n)$ where $t_0(n) \geq \log \epsilon(n)^{-1} + \log \delta(n)^{-1} + 3$, Algorithm 6.2 outputs $\mathrm{B}_0^{t(n)-1}(x)$ with probability at least $2^{-t(n)}(1 + \epsilon(n)/2)$.

Proof. For random x, $[2^{-t(n)}x]_N$ is uniformly distributed modulo N. To simplify expressions, let A be the event that Algorithm 6.2 outputs the correct value, and for $b \in \{0, 1\}$, $A(b)$ denotes the event that the algorithm is correct given that $\mathfrak{O}(E_N([2^{-t(n)}x]_N)) = \mathrm{bit}_i([2^{-t(n)}x]_N)$ *and* $\mathrm{bit}_i([2^{-t(n)}x]_N) = b$. Finally, for $b \in \{0, 1\}$ put

$$q_b \triangleq \Pr[\mathfrak{O}(E_N([2^{-t(n)}x]_N)) = \mathrm{bit}_i([2^{-t(n)}x]_N) \wedge \mathrm{bit}_i([2^{-t(n)}x]_N) = b]$$

and

$$p_b \triangleq \Pr[\mathfrak{O}(E_N([2^{-t(n)}x]_N)) = 1 \mid \mathrm{bit}_i([2^{-t(n)}x]_N) = b].$$

Then, by (6.1), we have $|p_1 - p_0| \geq \epsilon(n)$, and we may in fact assume that $p_1 - p_0 > 0$, otherwise we simply invert all outputs from $\mathfrak{O}$.

We have

$$Pr[A] \geq \Pr[A(0)]q_0 + \Pr[A(1)]q_1. \tag{6.2}$$

By definition,

$$\begin{aligned} q_b &= \Pr[\mathfrak{O}(E_N([2^{-t(n)}x]_N)) = \mathrm{bit}_i([2^{-t(n)}x]_N) \mid \mathrm{bit}_i([2^{-t(n)}x]_N) = b] \\ &\quad \cdot \Pr[\mathrm{bit}_i([2^{-t(n)}x]_N) = b] \end{aligned}$$

so since $[2^{-t(n)}x]_N$ is uniformly distributed in $\mathbb{Z}_N$, $q_0 = (1-p_0)\frac{1+\beta_i(N)}{2}$ and $q_1 = p_1\frac{1-\beta_i(N)}{2}$. Hence, continuing from (6.2) above,

$$\Pr[A] \geq \Pr[A(0)](1-p_0)\frac{1+\beta_i(N)}{2} + \Pr[A(1)]p_1\frac{1-\beta_i(N)}{2}.$$

Next, it is easy to see that for $b \in \{0,1\}$,

$$\Pr[A(b)] = \frac{1}{\#\mathfrak{J}} = \frac{1}{\#\{j \mid \exists z, 0 \leq z \leq 2^{i-t_0(n)} \wedge \mathrm{bit}_i([j2^{-t(n)} + z]_N) = b\}}.$$

This holds since given that the oracle is correct on deciding the ith bit, then $\mathfrak{J}$ does contain the correct choice for $\mathrm{B}_0^{t(n)-1}(x)$. Hence, as $[2^{-t(n)}x]_N$ is uniformly distributed in $\mathbb{Z}_N$, we have $\Pr[\mathrm{bit}_i([2^{-t(n)}x]_N) = 0] = (1+\beta_i(N))/2$, so for $b = 0$ for instance, one would expect $\#\mathfrak{J} = 2^{t(n)}(1+\beta_i(N))/2$. However, this is not completely true, but since $\Pr[\mathrm{bit}_i([j2^{-t(n)} + z]_N) = \mathrm{bit}_i([j2^{-t(n)}]_N)] = 1 - 2^{-t_0(n)}$, we certainly have $\#\mathfrak{J} \leq 2^{t(n)}((1+\beta_i(N))/2 + 2^{-t_0(n)})$. A similar statement hold when the ith bit is 1. Hence,

$$\begin{aligned} \Pr[A] &\geq 2^{-t(n)}\Big(\frac{1}{(1+\beta_i(N))/2 + 2^{-t_0(n)}}(1-p_0)\frac{1+\beta_i(N)}{2} \\ &\qquad + \frac{1}{(1-\beta_i(N))/2 + 2^{-t_0(n)}}p_1\frac{1-\beta_i(N)}{2}\Big) \\ &= 2^{-t(n)}\Big(\frac{1}{1+2^{-(t_0(n)-1)}/(1+\beta_i(N))}(1-p_0) \\ &\qquad + \frac{1}{1+2^{-(t_0(n)-1)}/(1-\beta_i(N))}p_1\Big) \\ &\geq 2^{-t(n)}\left(\frac{1}{1+2^{-(t_0(n)-1)}}(1-p_0) + \frac{1}{1+2^{-(t_0(n)-1)}\delta(n)^{-1}}p_1\right) \\ &\geq 2^{-t(n)}\left((1-2^{-(t_0(n)-1)})(1-p_0) + (1-2^{-(t_0(n)-1-\log\delta(n)^{-1})})p_1\right) \\ &= 2^{-t(n)}\left(1 + p_1 - p_0 - p_1 2^{-(t_0(n)-1-\log\delta(n)^{-1})} - (1-p_0)2^{-(t_0(n)-1)}\right) \\ &\geq 2^{-t(n)}\left(1 + \epsilon(n) - 2\cdot 2^{-(t_0(n)-1-\log\delta(n)^{-1})}\right) \\ &\geq 2^{-t(n)}\left(1 + \epsilon(n)/2\right), \end{aligned}$$

using the definition of $t_0(n)$ and that $0 \leq p_1, p_0 \leq 1$. □

Combining the above lemma, the proof in [1] of simultaneous security for the $t(n)$ least significant RSA bits, and our result in Theorem 5.22 now establishes the main result:

Theorem 6.4. For all non-negligible $\epsilon(n)$, any single bit in x is $\epsilon(n)$-secure for RSA, or else RSA can be broken in random polynomial time.

7 Simultaneous Security of RSA Bits

The notion of simultaneous security for RSA bits is, as mentioned, defined in terms of indistinguishability: a set of $d(n)$ bits, $\mathrm{B}_j^{j+d(n)-1}(x)$, is said to be secure if given $E_N(x)$, $\mathrm{B}_j^{j+d(n)-1}(x)$ is polynomially indistinguishable from a random string of the same length.

In [1], the simultaneous security for the $O(\log n)$ least significant bits of RSA follows more or less directly from the individual security of these bits. The proof uses Yao's *next-bit-test*, [30]: a function h, $\|h(x)\| = d$, is polynomially indistinguishable from the uniform distribution on $\{0,1\}^d$, if and only if, for each i, $1 \leq i \leq d-1$, $\mathrm{bit}_i(h(x))$ is secure, *given* $\mathrm{bit}_0(h(x)), \mathrm{bit}_1(h(x)), \ldots, \mathrm{bit}_{i-1}(h(x))$. Hence, assuming the existence of an oracle that given these bits predicts the ith, one essentially has an oracle for the ith bit. The only problem is to supply that oracle with $\mathrm{bit}_0(h(x)), \ldots, \mathrm{bit}_{i-1}(h(x))$. But when $h(x)$ is the d least significant RSA bits, this is a relatively easy task. One can assume that these bits of x are all zeros, so that when sampling the oracle, the value of these bits agree with the same bits of the added sample point: $[r_k x]_N$. These latter bits in turn, are known by a lemma similar to Lemma 4.2. Trying to apply the same method for the internal bits, we run into an obstacle. When j is far away from the end-bits, even if we assume that bits $j, \ldots, j+i-1$ of x and $[r_k x]_N$ are known, we do not know the value of these bits in the value supply to the oracle (which in our described method is of the form $[(r_k + 2^{-\tau})x]_N$), as the least significant bits of x causes wrap-around and unknown bits are shifted into the bit-segment we are considering. Thus we need to supply some of the bits we are trying to determine.

To remedy the problems involved, instead of taking the standard route via the next-bit-test, we use the well-known *Computational XOR-Lemma* by Vazirani and Vazirani, [28]. The following version is adopted from [12].

Lemma 7.1 (The Computational XOR-Lemma). Suppose that there is a pptm D such that

$$\left|\Pr[D(E_N(x), \mathrm{B}_j^{j+d(n)-1}(x)) = 1] - \Pr[D(E_N(x), R) = 1]\right| \geq \epsilon(n),$$

the probability taken over $x \in_{\mathcal{U}} \mathbb{Z}_N$, $R \in_{\mathcal{U}} \{0,1\}^{d(n)}$ and D's random choices (i.e. the two distributions are polynomially $\epsilon(n)$-distinguishable). Then there is a nonempty set $K \subset [j..j+d(n)-1]$ and a $\mathfrak{O}$ so that

$$\Pr\left[\mathfrak{O}(E_N(x), K) = \oplus_{k\in K}\,\mathrm{bit}_k(x)\right] \geq \frac{1}{2} + \frac{\epsilon(n)}{2^{d(n)}},$$

the probability taken over $x \in_{\mathcal{U}} \mathbb{Z}_N$, and $\mathfrak{O}$'s random choices.

Using this, we can prove

Theorem 7.2. Let $d(n) \in O(\log n)$. Then any set of $d(n)$ consecutive bits of x is simultaneously secure for RSA, or else RSA can be inverted in random polynomial time.

The idea is the same as before: Using an oracle for $\oplus_{k\in K}\,\mathrm{bit}_k(x)$, there are two possible paths to follow. We either decide bits two-by-two (the lsb and another bit, determined below), or, we find a transformation that converts the oracle into one that distinguishes intervals at distance $\frac{N+1}{2}$, enabling inversion through Lemma 5.4.

Proof. With K as in Lemma 7.1, let $i \triangleq \max_{k\in K} k$. We would like to decide $\mathrm{bit}_{i+1}(x), \mathrm{lsb}(x)$ so consider the $\Pi(N,i)$-plane as before and fix some $\alpha_N^{\tau(n)}(u,v)$.

First assume that $\alpha_N^{\tau(n)}(u,v)$ is nicely distributed modulo 2^{i+1} (i.e. there is no good, small rational approximation to $\alpha_N^{\tau(n)}(u,v)/2^{i+1}$). Looking back at the proof of Lemma 5.19 we see that all that we needed was that we had two sets where we knew that the oracle behaved differently and that not too many boxes were split among the two sets. In the current case the oracle predicts $\oplus_{k\in K}\,\mathrm{bit}_k(x)$. Now redefine $S^{(0)}, S^{(1)}$ from Definition 5.8, page 14, as the sets $S^{(b)} \triangleq \{x \mid \oplus_{k\in K}\,\mathrm{bit}_k(x) = b\}$, and notice that these two sets describe vertical stripes in the plane on which the oracle behaves differently. In addition, these stripes are of non-negligible width ($\geq 2^{i-d(n)}$) relative to 2^i. Making the division on the I-axis of the plane sufficiently fine-grained, we can make our boxes narrow enough so that not too many are split between stripes and by the properties of $\alpha_N^{\tau(n)}(u,v)$, the right fraction of boxes fall into $S^{(0)}, S^{(1)}$.

Secondly, assume that $\alpha_N^{\tau(n)}(u,v)/2^{i+1}$ is close to some r/s with s even (or that v is even, which is a similar case). We then have $s\alpha_N^{\tau(n)}(u,v) \approx r2^i$, r odd. By the choice of i, $\oplus_{k\in K}\,\mathrm{bit}_k(x) = \mathrm{bit}_i(x) \oplus (\oplus_{k\in K\setminus\{i\}}\,\mathrm{bit}_k(x))$, so that two values differing by 2^i (or an odd multiple thereof), differ also in $\oplus_{k\in K}\,\mathrm{bit}_k(x)$. Hence this case is treated similar to Lemma 5.20.

It remains to study the case when we have a good approximation with a small, odd denominator s. We do the same oracle conversion (by applying φ^{-1}) as before. What we need to show is that orbits at distance 2^i gets mapped by φ to values approximately at distance $N/2$ (which of course still holds) and that there are two orbits at distance 2^i where the original oracle behaves differently. Again, by the choice of i there must be two such orbits.

For the most significant bits the definition of simultaneous security in the case of biased bits. The defintion is an extension of the definition of the security of one bit. Given the definition the argument of Section 6 goes through virtually without change. Since the most significant bits of $[2^{-t_0}]_N$ are determined by the least significant bits of x, distinguishing the former from random bits is almost equivalent to distinguishing the latter from random bits. The details are very similar to argument for the individual bits and we again omit them. $\square$

8 Security of Rabin Bits

The Rabin encryption function is defined by $R_N(x) \triangleq [x^2]_N$ where $N = pq$ as before. Many of the earlier results for RSA (e.g. [28, 1]), carry over to the Rabin function in a straight-forward manner. One main complication to take care of is of basic nature, namely that R_N is not a 1–1 function since there are four roots to each quadratic residue. Hence, given some r, it is not well-defined what the "ith bit of $\sqrt{r}$" should be. One standard way to handle this problem is to demand $p \equiv q \equiv 3 \bmod 4$ (sometimes such N are called Blum-integers) and restrict the domain of R_N to

$$M_N \triangleq \{x \in \mathbb{Z}_N \mid x < N/2 \text{ and } (x/N) = 1\}$$

(where $(\cdot/N)$ denotes the Jacobi-symbol). It can then be shown that the function

$$R'_N(x) \triangleq \begin{cases} R_N(x), & \text{if } R_N(x) < N/2; \\ N - R_N(x), & \text{otherwise} \end{cases}$$

induces a permutation on M_N.

This approach runs into technical problems in our situation. When searching for boxes in $\Pi(N,i)$, where the oracle behaves differently on $S, S+\alpha_N^{\tau(n)}(u,v)$, we need that all of these boxes contain a non-negligible fraction of x with $(x/N) = 1$. Hence we need a result on the distribution of (x/N) in "rectilinear" subsets of $\mathbb{Z}_N$. There are related results known for the distribution of (x/p) (i.e. modulo primes) in *intervals*. These state that in $[z..z+L] \subset \mathbb{Z}_p$, the fraction of x with $(x/p) = 1$ (or -1) is very close to $\frac{1}{2}$, provided $L \geq \sqrt{p}$, see [6, 9] for instance. Notice now that a horizontal line (a "slice" of a box) in the plane corresponds to an interval modulo N of length $2^i/\operatorname{poly}(n)$. Since $(x/N) = (x/p)(x/q)$, it turns out that the distribution results mentioned are applicable as long as the width of our boxes is not too small (relative to N), for which it suffices that $i \geq 3n/4 + O(\log n)$. Similarly, when the height, h, of our boxes is large enough (when $i \leq n/4 - O(\log n)$), we can make a similar argument since vertical lines in the plane correspond to an arithmetic progression over a sub-interval to $\mathbb{Z}_N$: $\{x_0 + j2^{i+1} \mid j = 0, 1, \ldots, h-1\}$. Hence we claim, without going into the details, that this can be used prove security for roughly half of the bits.

Now, it seems very probable that, in fact, the equidistribution results of (x/N) actually holds also when both the width *and* the height of the boxes are small, as long as the *measure* of the box is non-negligible in comparison to N. Thus, under this conjecture, the results carry over to *all* bits.

However, we propose another way of converting the Rabin function. We drop the demand that $(x/N) = 1$. We then define on $M'_N \triangleq \{x \in \mathbb{Z}_N \mid x < N/2\}$:

$$R''_N(x) \triangleq R'_N(x), (x/N)$$

i.e we output the Jacobi symbol as well.

Our oracle for the ith bit now gets as input some $(z, b) \in M'_N \times \{-1, 1\}$ and supposedly (with advantage $\epsilon(n)$) answers by $\mathrm{bit}_i(x)$ where x is the unique root of z lying in M'_N and having $(x/N) = b$. When sampling the oracle, we now need to be able to supply the oracle with the Jacobi symbol of $[(r_j + 2^{-\tau(n)})x]_N$. But this is not difficult, since by the multiplicativity of $(\cdot/N)$, this is determined by (x/N) and $((r_j + 2^{-\tau(n)})/N)$, which we can compute. The only other concern is that when covering the plane by orbits of boxes, we must be aware that the oracle's advantage is for values in M'_N, i.e. the "lower half" of the plane. The interested reader may verify that all details can be taken care of.

Theorem 8.1. For each i, given $R''_N(x)$, $\mathrm{bit}_i(x)$ is secure, unless $R''_N(x)$ can be inverted in random polynomial time. Similarly, blocks of $O(\log n)$ bits of x are simultaneously secure.

9 Security of Discrete Log Bits

Let $f_{p,g}(x) = [g^x]_p$, p an n-bit prime and g a generator for $\mathbb{Z}_p^*$. Suppose that $p - 1 = p'2^k$, where p' is odd. Given $f_{p,g}(x)$, the k least significant bits of x are "easy" since they can be found by the Pohlig-Hellman algorithm, [23], and the $O(\log n)$ following bits are secure, see Peralta [22]. Also, the $O(\log n)$ most significant bits are secure; Long and Wigderson [19].

By a reduction from factoring Blum-integers $N = pq$ (and relaxing that g must generate all of $\mathbb{Z}_N^*$) Håstad, Schrift, and Shamir, [16], shows that all bits of x are individually hard with respect to $f_{N,g}(x)$, and $n/2$ bits are simultaneously secure. Patel and Sundaram, [21], adopt the techniques from [16] and prove that if $f_{p,g}$ is a one-way function, even if x is restricted to be "small", then almost all the bits of x are (simultaneously) hard. Using another bit-representation than the standard binary, Schnorr [25], recently proved security for all bits in this representation under similar assumptions.

Hence, despite the large attention given also to the bit security problem of $f_{p,g}(x)$, the (general) problem has remained open. Can our methods developed here be used to prove security for *all* bits of x? When trying to extend our method one immediate problem is encountered.

The problem is that we cannot query the oracle on $f_{p,g}(2^{-\tau}x)$ when the group order, $p - 1$, is even. By the work of Schnorr in [25], we can however reduce the problem to a subgroup of odd order, p'. We give a quick overview of this reduction. First, by the remark above, $u = [x]_{2^k}$ is easily found. The remaining bits of x can then be found as $\lfloor x/2^k \rfloor$, in other words as the discrete log of g^x/g^u, to the base g^{2^k}, and this value is considered modulo p'. Finally, notice that the ith bit of this number is just the $(i + k)$th bit of x.

Superficially one would expect the rest of the argument to go through. The only function specific property we need is that given $E_N(x)$ and a we can compute $E_N(ax)$. This is simply replaced by the fact that $g^{ax} = (g^x)^a$ which makes g^{ax} easily computable. A problem that was dealt with in one line in the RSA-case was the possible non-existence of φ^{-1}. If the inverse did not exist then we could factor N and immediately invert RSA. In the case of discrete logarithm we do not get such dramatic effects from the non-invertability of φ and we have to look more closely at this problem.

If φ is not invertible then $(sP_1 - \kappa, p') = d > 1$. When $d \in O(\text{poly}(n))$ we proceed as follows. We know that $\mathbb{Z}_{p'} \simeq \mathbb{Z}_d \times \mathbb{Z}_l$ where $l = p'/d$. Recall that we are in the situation where we would like to convert the ith bit oracle into one that distinguishes intervals at distance $(p'+1)/2$ by querying it on $\text{bit}_i(\varphi^{-1}(x))$ where $\varphi(x) = [(sP_1 - \kappa)x]_{p'}$ and using this we want to apply the method by Fischlin and Schnorr. Now, φ^{-1} exists only modulo l, but for z such that $[z]_d = 0$, i.e. $z = dz'$, we can define a pseudo-inverse by

$$\varphi^{-1}(z) \triangleq \mu_l[(sP_1 - \kappa)^{-1}z]_l + \mu_d r$$

where μ_l, μ_d are the Chinese remaindering coefficients $\mathbb{Z}_d \times \mathbb{Z}_l \to \mathbb{Z}_{p'}$ and where we choose r uniformly at random in $\mathbb{Z}_d$ each time we compute $\varphi^{-1}(z)$. This gives a uniformly distributed value in the inverse image of $z = dz'$ and some simple calculations shows that this pseudo-inverse retains the oracle's distinguishing advantage.

Rather than computing x, we now compute $[dx]_{p'}$ and also chose the pairwise independent points as multiples of d. In this case all values z supplied to the oracle satisfy $[z]_d = 0$ and we can use the pseudo-inverse above. This gives x modulo l, and x modulo d can be computed either by exhaustive search, or the Pohlig-Hellman algorithm. We then finally use the Chinese remainder theorem to obtain x modulo p'.

What remains is to analyze the probability that the gcd is large.

Lemma 9.1. Fix $t, w < t$ and let $p' = P_1 2^w + P_0$, be a randomly chosen t-bit integer (not necessarily a prime). Then

$$\Pr_{p'}[\exists s, \kappa \leq M \text{ s.t. } (sP_1 - \kappa, p') \geq D] \in O\left(\frac{M^2}{D} + tM^3 \max\left(2^{-w}, 2^{-(t-w)}\right)\right)$$

Proof. Say that p' is "bad" if there are $s, \kappa \leq M$ such that $(sP_1 - \kappa, p') \geq D$. Clearly, $(sP_1 - \kappa, p') \leq sP_1 + |\kappa| \leq 2M2^{t-w} \triangleq D_1$. Then

$$\begin{aligned} \Pr_{p'}[p' \text{ bad }] &\leq \sum_d \sum_{s,\kappa} \Pr_{p'}[(sP_1 - \kappa, p') = d] \\ &= \sum_d \sum_{s,\kappa} \underbrace{\sum_{P_1} \Pr_{P_0}[(sP_1 - \kappa, p') = d \mid P_1] \Pr[P_1],}_{(*)} \end{aligned} \tag{9.1}$$

where the sums range over $D \leq d \leq D_1$, $s, \kappa \leq M$ and $0 \leq P_1 < 2^{t-w}$. Next,

$$\begin{aligned}(*) \quad &= \sum_{P_1 : d | sP_1 - \kappa} \Pr_{P_0}[(sP_1 - \kappa, p') = d \mid P_1 \wedge d | sP_1 - \kappa] \Pr[P_1] \\ &\leq \sum_{P_1 : d | sP_1 - \kappa} \left(\frac{1}{d} + 2^{-w}\right) \Pr[P_1] = \left(\frac{1}{d} + 2^{-w}\right) \Pr_{P_1}[d | sP_1 - \kappa].\end{aligned}$$

Now, $d | sP_1 - \kappa$ if and only if $sP_1 \equiv \kappa \bmod d$, and this equation is solvable (in P_1) if and only if (d, s) divides κ, in which case there are precisely (d, s) solutions to $P_1 \bmod d$. Hence, since $\kappa \leq M$, for each fixed d, s, there are at most $M/(d, s)$ different κ possible, so continuing from (9.1),

$$\begin{aligned}\Pr_{p'}[p' \text{ bad }] \quad &\leq \sum_d \left(\frac{1}{d} + 2^{-w}\right) \sum_s (d, s) \sum_{\kappa : (d,s) | \kappa} \left(\frac{1}{d} + 2^{-(t-w)}\right) \\ &\leq \sum_d \left(\frac{1}{d} + 2^{-w}\right) \sum_s \left(\frac{M}{d} + \frac{M}{(d, s)} 2^{-(t-w)}\right) \\ &\leq \sum_d \sum_s \left(\frac{M}{d^2} + \frac{M}{d} 2^{-(t-w)} + \frac{M}{d} 2^{-w} + M 2^{-t}\right) \\ &\leq M^2 \sum_d \left(\frac{1}{d^2} + \frac{1}{d}(2^{-(t-w)} + 2^{-w}) + 2^{-t}\right),\end{aligned}$$

and this sum is bounded by $O(M^2(D^{-1} + \log D_1 \max(2^{-w}, 2^{-(t-w)}) + D_1 2^{-t}))$. □

Theorem 9.2. Unless the discrete log problem can be solved in random polynomial time, with probability $1 - O(n^{-1})$ over random choices of $p = p' 2^k + 1$, $\|p\| = n$, bits $k, \ldots, n-1$ of x are individually secure for $f_{p,g}(x)$. Blocks of $O(\log n)$ bits are simultaneously secure.

Proof. Let $i_0(n) \triangleq 5 \log n + 6 \log \epsilon(n)^{-1}$, where $\epsilon(n)$ is the assumed oracle-advantage. Also, define $M \triangleq cn\epsilon(n)^{-2}$ for a constant c, and $D \triangleq n^5 \epsilon(n)^{-4}$.

Choose a random n-bit number p (not necessarily a prime), and let $k \geq 0$ be the highest power of 2, dividing $p-1$. Consider a fixed $i \in [k+i_0(n)..n-1-i_0(n)]$ (by the results in [19, 22], these are the interesting bits). Write $p = p'_i 2^k + 1$ as above and call p "bad" for this i if $p'_i = P_1 2^{i+1-k} + P_0$ is bad in the sense of Lemma 9.1, i.e. if there are $s, \kappa \leq M$ (by Lemma 5.26, M as above suffices) such that $(sP_1 - \kappa, p'_i) \geq D$. By Lemma 9.1 with $t = n - i$, $w = i - k$, p is bad with probability $O(M^2 D^{-1} + tM^3 \max(2^{-w}, 2^{-(t-w)}))$, which by the choices above is $O(n^{-3})$. Moreover, there are less than n different bit positions, i, to consider, so the probability that one of them gives a bad p'_i is $O(n^{-2})$.

What does this tell us about the probability that p is bad when p is a prime? The worst case is clearly if all bad ps are prime numbers. By the prime number theorem, the probability that an n-bit integer is a prime is $\Theta(n^{-1})$. Thus,

$$\Pr_p[p \text{ is a bad prime }] \leq \frac{\Pr_{p \in_{\mathcal{U}} \mathbb{Z}_{2^n}}[\exists i \text{ s.t. } p'_i \text{ is bad }]}{\Pr_{p \in_{\mathcal{U}} \mathbb{Z}_{2^n}}[p \text{ is prime }]}.$$

We may thus loose at most an extra factor of n here, but the probability that p is a bad prime is still bounded by $O(n^{-1})$. Finally if p is not a bad prime the results of the previous sections extend to show that all bits are secure. □

It remains to extend Theorem 9.2 to cover all values of p and in particular to treat the case when the above gcd is large. Although this might sound like a technicality, it seems that such an extension would require new techniques. To see this, consider the following example.

Assume that $p = q(2^{i+1} + 2) + 1$ where q is a prime of size around $2^{i/2}$. Our bit security proofs compute the discrete logarithm of a number y by querying the ith bit of the discrete logarithm of numbers of the form $y^a g^b$. This is equivalent to reconstructing x from information on the ith bit of $ax + b$. Now we claim that using this approach, for the above p, it is hard to distinguish x and $x' = x + t(2^{i+1} + 2)$ for any $t > 0$. The reason is simply that $ax + b$ and $ax' + b$ (modulo $p - 1$) differ by $at(2^{i+1} + 2)$ and since at is only considered modulo q, except with exponentially small probability, the two numbers have the same value for their ith bit.

10 Security of $ax + b$ modulo p

As described in the introduction, the methods utilized in this paper were first discovered when completing the proof of the results claimed in [20]. We here give the proofs for this original application in a slightly stronger form. The results are stronger in that they apply to smaller primes. We are interested in the following family of hash functions.

Definition 10.1. Let H_m be the set of functions of the form $h(x) \triangleq ax + b \bmod p$ with the following probability distribution. The number p is a random prime of m bits while a and b are random numbers modulo p.

We need to be define a family of hard core predicates.

Definition 10.2. A family B of predicates is hard core for a one-way function f if given $f(x)$ and a description of a random $b \in B$, $b(x)$ cannot be predicted with a non-neglible advantage. The definition extends to functions outputting more than 1 bit by requiring that the output cannot be distinguished from random bits with non-negligible advantage.

Theorem 10.3. *Let f be any one-way function and $m = \omega(\log n)$. Then for any i, $0 \leq i < m$ and any constant c, $\mathrm{B}_i^{i+c\log n}(H_m)$ form a family of hard core functions for f.*

Proof. Most of the proof is identical to the previous proofs with the following syntactical difference. In previous situation we created encryptions of numbers of the form ax from encryptions on x. In the current situation this is not possible since we have no structure in f. The point is that we are getting predictions on bits of $ax + b$ and this number can be manipulated by changing a and b

which are at our disposal. In particular, division by 2 can be accomplished by replacing (a, b) by $(a/2, b/2)$. Thus we are in essentially the same situation as before.

Assume that we have some $\mathfrak{O}$ that predicts the ith bit of $ax + b$ modulo p given $f(x)$, a, b and p with non-negligible advantage $\epsilon(n)$ and we want to recover x. Let us first fix an x such that the advantage over random a, b and p is at least $\epsilon(n)/2$. Let us say that a p is *good* for this x if the advantage of $\mathfrak{O}$ for this fixed p (over random a and b) is at least $\epsilon(n)/4$. It is easy to see that at least a fraction $\epsilon(n)/4$ of all p are good.

Let us see how the methods from Section 5 and Section 6 extend to compute x modulo p for good p. None of the problems encountered in previous extensions show up. The function is 1-1 and the modulus is prime and hence it is easy to divide by 2 and invert φ. However, if $m < n$ we cannot check the result. This implies that the polynomial number of different guesses for x modulo p that are given by the polynomially many different choices in the construction of our pairwise independent sample points cannot be immediately distinguished. The following powerful result of Goldreich, Ron, and Sudan [13] comes to our rescue.

Theorem 10.4. Let $p_1 < p_2 < p_3 \ldots < p_s$ be primes, t and k be integers and $(r_1)_{j=1}^{s}$ be given numbers. Then, provided

$$t \geq \Omega\left(\sqrt{ks\frac{\log p_s}{\log p_1}}\right),$$

it is possible in polynomial time to output the list of all numbers z such that $0 \leq z \leq \prod_{j=1}^{k} p_i$ and such that $z \equiv r_j$ modulo p_j for at least t different values of j.

To apply this theorem we proceed as follows. Let ℓ be a parameter to be specified shortly. Take ℓ different and random p_j each with m bits and apply the procedure equivalent of Section 5 and Section 6 to get a list of size $m^{c_1}\epsilon(n)^{-c_2}$ (for some constants c_1 and c_2 implicit in those proofs) of possible candidates for x modulo p_j for each p_j. Now for each j randomly pick one element r_j in the list and input the list of $(p_j)_{j=1}^{\ell}$ and $(r_j)_{j=1}^{\ell}$ to the algorithm existing by Theorem 10.4. For any element z output by that procedure compute $f(z)$ to see whether z is an acceptable answer.

We need to specify the choice of k and t. Since x has n bits we have $x \leq 2^n$ and since each p_j is at least 2^{m-1} we can have $k = \lceil \frac{n}{m-1} \rceil$. Let us estimate the number of modular equations satisfied by x. First the fraction of p_j that are good is at least $\epsilon(n)/4$ and as stated above for each such p_j we have a list of length $m^{c_1}\epsilon(n)^{-c_2}$ such that with probability at least 1/2 the value of x modulo p_j appears on it. Thus the expected number of modular equations satisfied by x is at least

$$\ell m^{-c_1}\epsilon(n)^{c_2+1}/8.$$

For sufficiently large ℓ with probability at least 1/2 the actual number is at least half the expected value i.e.

$$\ell m^{-c_1} \epsilon(n)^{c_2+1}/16$$

and this is the value we choose for t. We need to check the condition of the Theorem 10.4 i.e. that

$$t \geq \Omega\left(\sqrt{ks\frac{\log p_s}{\log p_1}}\right),$$

which in our case is translates to

$$\ell m^{-c_1} \epsilon(n)^{c_2+1}/16 \geq \Omega(\sqrt{\ell n/m})$$

or

$$\ell \geq \Omega\left(m^{2c_1-1} n \epsilon(n)^{-2(c_2+1)}\right).$$

This implies that we can choose an ℓ of polynomial size which satisfies this inequality and in this case the procedure runs in polynomial time and recovers x with probability $1/2$. We conclude that when f is a one-way function such an oracle cannot exist and the ith bit is secure.

The extension to simultaneous security runs along the usual lines. □

11 Discussion and Open Problems

Although the reduction from RSA inversion to predicting the individual bits is polynomial time, it is still quite complex and it is hard to give practical implications of the results obtained here. It would therefore be of great interest to find, if possible, a simpler proof, leading to tighter relation between bit security and overall security for RSA.

Hence, to hide partial information on x in a practical application involving RSA, it is of course still wise to use RSA in a more sophisticated way such as in [2].

For the simultaneous security, it is in general impossible to go beyond $O(\log n)$ bits. For specific functions (e.g. [16]) it has been done, so we ask if it is possible also for RSA.

References

[1] Werner Alexi, Benny Chor, Oded Goldreich, and Claus P. Schnorr. RSA and Rabin functions: Certain parts are as hard as the whole. *SIAM Journal on Computing*, 17(2):194–209, 1988.

[2] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *Advances in Cryptology—Eurocrypt '94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111, May 9–12 1994, Perugia, Italy, 1995. Springer-Verlag.

[3] Michael Ben-Or, Benny Chor, and Adi Shamir. On the cryptographic security of single RSA bits. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, pages 421–430, Apr. 25–27 1983, Boston, Massachusetts, 1983. ACM.

[4] T. Beth, N. Cot, and I. Ingemarsson, editors. *Advances in Cryptology: Proceedings of Eurocrypt '84*, volume 209 of *Lecture Notes in Computer Science*, Apr. 9–11 1984, Paris, France, 1985. Springer-Verlag.

[5] Dan Boneh and Ramarathnam Venkatesan. Breaking RSA may not be equivalent to factoring. In Kaisa Nyberg, editor, *Advances in Cryptology—Eurocrypt '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 59–71, May 31–Jun. 4 1998, Espoo, Finland, 1998. Springer-Verlag.

[6] David A. Burgess. The distribution of quadratic residues and non-residues. *Mathematika*, 4:106–112, 1957.

[7] Ben Chor. *Two Issues in Public Key Cryptography.* ACM doctoral dissertation award. MIT Press, 1986.

[8] Benny Chor and Oded Goldreich. RSA/Rabin least significant bits are $\frac{1}{2} + \frac{1}{\mathrm{poly}(\log n)}$ secure. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology: Proceedings of CRYPTO '84*, volume 196 of *Lecture Notes in Computer Science*, pages 303–313, Aug. 19–22 1984, University of California, Santa Barbara, 1985. Springer-Verlag.

[9] Harold Davenport. On the distribution of quadratic residues $\pmod{p}$. *J. London Math. Soc.*, 8:46–52, 1933.

[10] Roger Fischlin and Claus P. Schnorr. Stronger security proofs for RSA and Rabin bits. In Walter Fumy, editor, *Advances in Cryptology—Eurocrypt '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 267–279, May 11–15 1997, Konstanz, Germany, 1997. Springer-Verlag.

[11] Oded Goldreich. On the number of close-and-equal pairs of bits in a string (with applications on the security of RSA's L.S.B.). In Beth et al. [4], pages 127–141.

[12] Oded Goldreich. The computational XOR-lemma—an exposition. Manuscript, 1991.

[13] Oded Goldreich, Dana Ron, and Madhu Sudan. Chinese remaindering with errors. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, to be held May 1–4 1999, Atlanta, Georgia, 1999. ACM.

[14] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.

[15] Shafi Goldwasser, Silvio Micali, and Po Tong. Why and how to establish a private code on a public network (Extended abstract). In *23rd Annual Symposium on Foundations of Computer Science* [17], pages 134–144.

[16] Johan Håstad, Avital W. Schrift, and Adi Shamir. The discrete logarithm modulo a composite hides $O(n)$ bits. *Journal of Computer and System Sciences*, 47:850–864, 1993.

[17] IEEE. *23rd Annual Symposium on Foundations of Computer Science*, Nov. 3–5 1982, Chicago, Illinois, 1982.

[18] Lauwerens Kuipers and Harald Niederreiter. *Uniform Distribution of Sequences.* Pure & Applied Mathematics. John Wiley & Sons, 1 edition, 1974.

[19] Douglas L. Long and Avi Wigderson. The discrete log hides $O(\log n)$ bits. *SIAM Journal on Computing*, 17(2):413–420, 1988.

[20] Mats Näslund. All bits in $ax + b \bmod p$ are hard. In Neal Koblitz, editor, *Advances in Cryptology—CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 114–128, Aug. 18–22 1996, University of California, Santa Barbara, 1996. Springer-Verlag.

[21] Sarvar Patel and Ganapathy S. Sundaram. An efficient discrete log pseudo random generator. In Hugo Krawczyk, editor, *Advances in Cryptology—CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 304–317, Aug. 23–27 1998, University of California, Santa Barbara, 1998. Springer-Verlag.

[22] René Peralta. Simultaneous security of bits in the discrete log. In Franz Pichler, editor, *Advances in Cryptology—Eurocrypt '85*, volume 219 of *Lecture Notes in Computer Science*, pages 62–72, Apr. 1985, Linz, Austria, 1986. Springer-Verlag.

[23] Stephen C. Pohlig and Martin Hellman. An improved algorithm for computing logarithms over GF(p). *IEEE Transactions on Information Theory*, IT-24(1):106–110, 1978.

[24] Ronald L. Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[25] Claus P. Schnorr. Security of almost all discrete log bits, 1998. Electronic Colloquium on Computational Complexity, report TR98-033. Available online from http://www.eccc.uni-trier.de/eccc/.

[26] Claus P. Schnorr and Werner Alexi. RSA-bits are $0.5 + \epsilon$ secure. In Beth et al. [4], pages 114–128.

[27] Avital W. Schrift and Adi Shamir. On the universality of the next bit test. In A. J. Menezes and S. A. Vanstone, editors, *Advances in Cryptology—CRYPTO '90*, volume 537 of *Lecture Notes in Computer Science*, pages 394–408, Aug. 11–15 1990, University of California, Santa Barbara, 1991. Springer-Verlag.

[28] Umesh V. Vazirani and Vijay V. Vazirani. Efficient and secure pseudo-random number generation (Extended abstract). In *25th Annual Symposium on Foundations of Computer Science*, pages 458–463, Oct. 24–26 1984, Singer Island, Florida, 1984. IEEE.

[29] Umesh V. Vazirani and Vijay V. Vazirani. RSA bits are .732 + ϵ secure. In David Chaum, editor, *Advances in Cryptology: Proceedings of CRYPTO '83*, pages 369–375, Aug. 22–24 1983, University of California, Santa Barbara, 1984. Plenum Press, New York and London.

[30] Andrew C. Yao. Theory and applications of trapdoor functions (Extended abstract). In *23rd Annual Symposium on Foundations of Computer Science* [17], pages 80–91.

A The Discrepancy of a Rational Sequence

This section follows closely the ideas behind the proof of Theorem 2.5 in [18]. The aim is to prove Theorem 5.25.

Definition A.1. Recall that for $\zeta \in \mathbb{Q}$, $[\zeta]_1$ denotes the fractional part, $\zeta \pmod 1$ and $\langle\zeta\rangle$ is the distance to the closest integer $\langle\zeta\rangle \triangleq \min([\zeta]_1, 1-[\zeta]_1)$. By a *rational sequence* we mean a sequence of the form $\{[j\zeta]_1 \mid 0 \le j \le T-1\}$ where $\zeta \in \mathbb{Q}$, $T \in \mathbb{N}$. We denote such a sequence by $(\zeta)_T$.

For any sequence $W_T = w_1, w_2, \dots, w_T \subset [0,1]$, the *discrepancy* of W is defined to be

$$\mathfrak{D}(W_T) \triangleq \sup_{0\le a<b<1} \left| \frac{\#(W_T \cap [a,b])}{T} - (b-a) \right|.$$

Our objective is first to prove the following theorem, from which the desired result then easily follows.

Theorem A.2. *If $\zeta \in \mathbb{Q}$ is of (Q,ψ)-type, then the rational sequence $(\zeta)_T$ satisfies*

$$\mathfrak{D}((\zeta)_T) \le 6\left(\frac{2}{Q} + \frac{8\psi \log^2 Q}{T}\right).$$

In order to do so, we first need a few preliminaries.

Theorem A.3 (Erdős-Turán). For *any* finite set $W_T = \{w_1, w_2, \dots, w_T\}$ of real numbers and *any* positive integer m:

$$\mathfrak{D}(W_T) \leq 6 \left(\frac{1}{m} + \sum_{h=1}^{m} \frac{1}{h} \left| \frac{1}{T} \sum_{j=1}^{T} e^{2\pi i h w_j} \right| \right).$$

A proof can be found in [18].

Lemma A.4. If $\zeta \in \mathbb{Q}$ is of (Q, ψ)-type, then for any $m \leq Q$:

$$\mathfrak{D}((\zeta)_T) \leq 6 \left(\frac{1}{m} + \frac{1}{T} \sum_{h=1}^{m} \frac{1}{h\langle h\zeta \rangle} \right).$$

Proof. By the Erdős-Turán Theorem,

$$\mathfrak{D}((\zeta)_T) \leq 6 \left(\frac{1}{m} + \sum_{h=1}^{m} \frac{1}{h} \left| \frac{1}{T} \sum_{j=1}^{T} e^{2\pi i h j \zeta} \right| \right)$$

for any m. Now,

$$\left| \sum_{j=1}^{T} e^{2\pi i h j \zeta} \right| \leq \frac{2}{|e^{2\pi i h \zeta} - 1|} = \frac{1}{|\sin \pi h \zeta|}$$

since $h\zeta$ is never an integer for $h \leq m \leq Q$. This also implies that $|\sin \pi h\zeta| = \sin \pi \langle h\zeta \rangle$. Finally, note that $\sin \pi x \geq 2x$ for $0 \leq x \leq 1/2$ so that

$$\frac{1}{|\sin \pi h\zeta|} = \frac{1}{\sin \pi \langle h\zeta \rangle} \leq \frac{1}{2\langle h\zeta \rangle}.$$

□

Lemma A.5. Suppose $\zeta \in \mathbb{Q}$ is of (Q, ψ)-type and $m \leq Q/2$. Then

$$\sum_{j=1}^{m} \frac{1}{j\langle j\zeta \rangle} \leq 8\psi \log^2 m.$$

Proof. Define $s_j = \sum_{k=1}^{j} 1/\langle k\zeta \rangle$, $j = 1, 2, \dots, m$. Then, by induction, it is easy to see that

$$\sum_{j=1}^{m} \frac{1}{j\langle j\zeta \rangle} = \sum_{j=1}^{m} \frac{s_j}{j(j+1)} + \frac{s_m}{m+1}. \tag{A.1}$$

If $0 \leq r < s \leq j \leq m \leq Q/2$,

$$\langle s\zeta \pm r\zeta \rangle = \langle (s \pm r)\zeta \rangle \geq \frac{1}{(s \pm r)\psi} \geq \frac{1}{2j\psi}$$

and hence

$$|\langle s\zeta\rangle - \langle r\zeta\rangle| \geq \frac{1}{2j\psi}. \tag{A.2}$$

Consider the intervals

$$\left[0, \frac{1}{2j\psi}\right), \left[\frac{1}{2j\psi}, \frac{2}{2j\psi}\right), \ldots, \left[\frac{j}{2j\psi}, \frac{j+1}{2j\psi}\right).$$

Each of these can by $(A.2)$ contain at most one rational of the form $\langle k\zeta\rangle$, $1 \leq k \leq j$, with no such in the first interval. Therefore

$$s_j = \sum_{k=1}^{j} \frac{1}{\langle k\zeta\rangle} \leq \sum_{k=1}^{j} \frac{2j\psi}{k} \leq 4j\psi \log j$$

so that from $(A.1)$,

$$\sum_{j=1}^{m} \frac{1}{j\langle j\zeta\rangle} \leq 4\psi \left(\sum_{j=1}^{m} \frac{\log j}{j} + \log m \right) \leq 8\psi \log^2 m.$$

□

We are now ready to prove Theorem A.2.

Proof of Theorem A.2. By Lemma A.4 and A.5, setting $m = Q/2$:

$$\mathfrak{D}((\zeta)_T) \leq 6 \left(\frac{1}{m} + \frac{1}{T} \sum_{j=1}^{m} \frac{1}{j\langle j\zeta\rangle} \right) \leq 6 \left(\frac{2}{Q} + \frac{1}{T} 8\psi \log^2(Q/2) \right).$$

□

Finally, we prove Theorem 5.25.

Proof of Theorem 5.25. Let $\zeta = \tilde{\alpha}_N^{\tau(n)}(u,v)/2^{i+1}$ and

$$p' = \Pr_{j \in_U \mathbb{Z}_{2^{\tau(n)}}} [a \leq [j\tilde{\alpha}_N^{\tau(n)}(u,v)]_{2^{i+1}} \leq b.]$$

Then

$$\begin{aligned} p' &= \Pr_j \left[\frac{a}{2^{i+1}} \leq j\zeta \bmod 1 \leq \frac{b}{2^{i+1}} \right] \\ &= \frac{\#\left(\{[j\zeta]_1 \mid 0 \leq j \leq 2^{\tau(n)} - 1\} \cap \left[\frac{a}{2^{i+1}}, \frac{b}{2^{i+1}}\right] \right)}{2^{\tau(n)}} \in \frac{b-a}{2^{i+1}} \pm \mathfrak{D}((\zeta)_{2^{\tau(n)}}). \end{aligned}$$

Since ζ is of $(Q(n), \psi(n))$-type, Theorem A.2 tells us that

$$\mathfrak{D}((\zeta)_{2^{\tau(n)}}) \leq 6 \left(\frac{2}{Q(n)} + \frac{1}{2^{\tau(n)}} 8\psi(n) \log^2(Q(n)) \right).$$

However, we are restricted to picking j in $\{0, \ldots, 2^{\tau(n)} - 2\}$ only. But it is easy to see that by omitting the single value $(2^{\tau(n)} - 1)\zeta$, this can only make the discrepancy go up by $2^{-\tau(n)}$ so certainly, if we pick j at random in $\{0, \ldots, 2^{\tau(n)} - 2\}$,

$$\left| \Pr_j[a \leq [j\tilde{\alpha}_N^{\tau(n)}(u,v)]_{2^{i+1}} \leq b] - \frac{b-a}{2^{i+1}} \right| \leq 7 \left(\frac{2}{Q(n)} + \frac{8\psi(n)\log^2(Q(n))}{2^{\tau(n)}} \right).$$

□