The Efficiency and Security of a Real Quadratic Field Based-Key Exchange Protocol

M. J. Jacobson, Jr., R. Scheidler and H. C. Williams

December 21, 2001

Abstract

Most cryptographic key exchange protocols make use of the presumed difficulty of solving the discrete logarithm problem (DLP) in a certain finite group as the basis of their security. Recently, real quadratic number fields have been proposed for use in the development of such protocols. Breaking such schemes is known to be at least as difficult a problem as integer factorization; furthermore, these are the first discrete logarithm based systems to utilize a structure which is not a group, specifically the collection of reduced ideals which belong to the principal class of the number field. For this structure the DLP is essentially that of determining a generator of a given principal ideal.

Unfortunately, there are a few implementation-related disadvantages to these schemes, such as the need for high precision floating point arithmetic and an ambiguity problem that requires a short, second round of communication. In this paper we describe work that has led to the resolution of some of these difficulties. Furthermore, we discuss the security of the system, concentrating on the most recent techniques for solving the DLP in a real quadratic number field.

1 Introduction and Motivation

In 1976, W. Diffie and M. Hellman introduced their by now legendary protocol for communicating a secret cryptographic key across an insecure channel. In their original scheme, two communication partners first agree on a large prime p and a nonzero element $g \in \mathbb{F}_p$; p and g are public. The two parties generate respective secret exponents a and b, and subsequently exchange g^a and g^b. From their own secret exponent and the information received from the other communicant, each party can now generate the secret common key g^{ab}. The only known attack on this scheme is for an eavesdropper to solve an instance of the *discrete logarithm problem* (DLP) in $\mathbb{F}_p$: given $g, g^a \in \mathbb{F}_p$, find a.

It was soon recognized that the Diffie-Hellman protocol could be extended to any sufficiently large group with fast arithmetic and a difficult discrete logarithm problem. Many such groups were suggested, including arbitrary finite fields, matrices over a finite field [15], the unit group of $\mathbb{Z}/n\mathbb{Z}$ with n a product of two primes [13], the group of points on an elliptic curve [14, 8], the Jacobian of a hyperelliptic curve [9], and the class group of an imaginary quadratic number field [3]. More careful

inspection of the scheme reveals that only a semi-group structure is required, and two non-group versions were proposed, using as underling structure the infrastructure of a real quadratic number field and function field, respectively [17, 18].

In order to conduct key exchange in the set $\mathcal{R}$ of reduced principal ideals of a real quadratic number field K, one requires an operation under which $\mathcal{R}$ is closed and associative. The most natural way to do this is to assign to two reduced principal ideals the reduced principal ideal "closest" to their (generally nonreduced) product; that is, the unique reduced ideal whose generator is closest to the product of the two respective generators. In this model, a reduced principal ideal $\mathfrak{r}$ is publicized beforehand. Alice and Bob generate respective secret exponents a and b and compute the reduced principal ideal closest to $\mathfrak{r}^a$ and $\mathfrak{r}^b$, respectively. The common key the unique reduced principal ideal closest to $\mathfrak{r}^{ab}$.

Unfortunately, ideal generators are generally too large to allow for efficient arithmetic. Moreover, in practice, they are only given up to a limited precision, so it may be impossible to determine the unique reduced principal ideal closest to a given ideal. The arithmetic of [17] opted for computing two possible candidates – adjacent reduced principal ideals – for the closest ideal. To make sure that both parties obtain the same two candidates for the final key ideal, generators had to be given to p correct bits where $2^p > 3072\sqrt{D}B^2$ and B is an upper bound on the exponents a and b. The problem of ambiguity was eliminated in [4], maintaining essentially the same precision p for computing the key ideal. In this paper, we give a more elegant version of the protocol of [17] which achieves five-fold ambiguity with considerably less precision, namely $2^p > 46B^2 \log_2 B$; note that this bound is independent of the radicand D of the field. In practice, the two communicants always obtain the same ideal after their exchange, but for the paranoid user, we describe how communicating five more bits guarantees uniqueness.

Our paper is organized as follows. In the next section, we review the basics of real quadratic number fields and their ideals. Section 3 introduces the so called (f, p) representations used to represent ideals and the algorithms underlying our protocol. In section 4, we present the actual key exchange protocol. In the remaining sections we discuss implementation and security issues.

2 Real Quadratic Number Fields

In this section we will review some basic properties of real quadratic number fields from a computational perspective. This material can be found in greater detail in [21] and [16].

Let D be a positive squarefree integer and let $K = \mathbb{Q}(\sqrt{D})$ be the real quadratic number field of discriminant $\Delta = (4/\sigma^2)D$ where $\sigma = 2$ if $D \equiv 1 \pmod 4$ and $\sigma = 1$ otherwise. For an element $\alpha = a + b\sqrt{D} \in K$ with $a, b \in \mathbb{Q}$, the conjugate of α is $\overline{\alpha} = a - b\sqrt{D}$. The maximal order of K is $\mathcal{O} = \mathbb{Z}[\omega]$ where $\omega = (\sigma - 1 + \sqrt{D})/\sigma$.

Every nonzero ideal $\mathfrak{a}$ in $\mathcal{O}$ is a $\mathbb{Z}$-module of rank 2 with a $\mathbb{Z}$-basis of the form $\{SQ/\sigma, S(P+\sqrt{D})/\sigma\}$ with $S, Q, P \in \mathbb{Z}$, $Q > 0$, $\sigma \mid Q$ and $\sigma Q \mid D - P^2$. If $\mathfrak{a}$ is primitive, then $S = 1$; in this case, write $\mathfrak{a} = (Q, P)$ for brevity. A primitive ideal $\mathfrak{a} = (Q, P)$ is reduced if Q is a minimum in $\mathfrak{a}$; that is, no nonzero $\alpha \in \mathfrak{a}$ satisfies $|\alpha| < Q$ and $|\overline{\alpha}| < Q$. If $\mathfrak{b}_1 = (Q_0, P_0)$ is any primitive ideal, then the

recursion

$$q_{j-1} = \left\lfloor \frac{P_{j-1} + \sqrt{D}}{Q_{j-1}} \right\rfloor, \quad P_j = q_{j-1}Q_{j-1} - P_{j-1}, \quad Q_j = \frac{D - P_j^2}{Q_{j-1}}, \tag{2.1}$$

$j \in \mathbb{N}$, defines a sequence of pairwise equivalent ideals $\mathfrak{b}_j = (Q_{j-1}, P_{j-1})$ $(j \in \mathbb{N})$. This, of course, is the continued fraction expansion of $(P + \sqrt{D})/Q$, where the q_j values are the *partial quotients* and the $(P_j + \sqrt{D})/Q_j$ values are the *complete quotients*. Here, $\mathfrak{b}_j = (\Psi_j)\mathfrak{b}_1$ where

$$\Psi_j = \prod_{i=1}^{j-1} \psi_i \text{ and } \psi_i = \frac{P_i + \sqrt{D}}{Q_{i-1}} \quad (1 \le i \le j-1). \tag{2.2}$$

for $j \in \mathbb{N}$. It is easy to see that $\Psi_{j+2} = q_j\Psi_{j+1} + \Psi_j$ and $\Psi_j\overline{\Psi_j} = (-1)^{j-1}Q_{j-1}/Q_0$ for $j \in \mathbb{N}$. If $\mathfrak{b}_1$ is reduced, then $\mathfrak{b}_j$ is reduced for all $j \in \mathbb{N}$, in which case

$$1 + \frac{1}{\sqrt{\Delta}} < \psi_j < \sqrt{\Delta} \text{ and } \psi_{j+1}\psi_j > 2 \quad (j \in \mathbb{N}). \tag{2.3}$$

If $\mathfrak{b}_1$ is nonreduced, then $\mathfrak{b}_j$ is reduced as soon as $Q_{j-1} > 0$ and $P_{j-1} < \sqrt{D}$; this happens after no more than $O(\log(Q_0/\sqrt{D}))$ iterations of (2.1). Furthermore, if $\mathfrak{b}_j$ is the first reduced ideal in the sequence $(\mathfrak{b}_i)_{i\in\mathbb{N}}$, then $\Psi_j \le 1$ (see Corollary 4.6 of [16]) and $|\overline{\Psi_i}| < 1$, $0 < P_i < \sqrt{D}$, and $\sigma \le Q_i < 2\sqrt{D}$ for $i \ge j$. The process of obtaining $\mathfrak{b}_{j+1}$ from $\mathfrak{b}_j$ is called a *forward step*. Also, by the symmetry property of the continued fraction of ω, we see that ψ_i is a complete quotient in that continued fraction.

Suppose $\mathfrak{b}_{j+1} = (Q_j, P_j)$ is reduced for some $j \in \mathbb{N}$. Then one can perform a *backward step* to obtain $\mathfrak{b}_j = (Q_{j-1}, P_{j-1})$ from the recursion

$$Q_{j-1} = \frac{D - P_j^2}{Q_j}, \quad q_{j-1} = \left\lfloor \frac{P_j + \sqrt{D}}{Q_{j-1}} \right\rfloor, \quad P_{j-1} = q_{j-1}Q_{j-1} - P_j. \tag{2.4}$$

If we set $\mathfrak{b}_1^* = (Q_0^*, P_0^*) = \mathfrak{b}_j$, so $Q_0^* = Q_{j-1}$ and $P_0^* = P_{j-1}$, then (2.4) generates a sequence of reduced ideals $\mathfrak{b}_i^* = (Q_{i-1}^*, P_{i-1}^*)$ $(i \in \mathbb{N})$. Here, we have $\mathfrak{b}_i^* = (\Psi_i^*)\mathfrak{b}_1^*$ where $\Psi_i^* = \Psi_{j-i+1}$ and in fact $\mathfrak{b}_i^* = \mathfrak{b}_{j-i+1}$. Analogous to (2.2), we have

$$\Psi_i^* = \prod_{j=1}^{i-1} \psi_j^* \text{ where } \psi_j^* = \frac{\sqrt{D} - P_{j-1}^*}{Q_{j-1}^*} = \frac{1}{\psi_{i-j}} \quad (1 \le j \le i-1) \tag{2.5}$$

as well as $|\Psi_i^*\overline{\Psi_i^*}| = Q_{i-1}^*/Q_0$ and $\Psi_{i+2}^* = \Psi_i^* - q_i^*\Psi_{i+1}^*$ for $1 \le i \le j-2$.

Every principal ideal $\mathfrak{a}$ has a unique generator θ such that $1 \le \theta < \epsilon$ where ϵ is the unique fundamental unit of K with $\epsilon > 1$. For $i \in \mathbb{N}$, let $\mathfrak{b}_i = (\Psi_i)$ with $\mathfrak{b}_1 = \mathcal{O}$. If $\mathfrak{a}$ is nonreduced, then there exists a unique index $j \in \mathbb{N}$ with $\Psi_j < \theta < \Psi_{j+1}$. We define $\rho(\mathfrak{a}) = \mathfrak{b}_{j+1}$ and $\rho^{-1}(\mathfrak{a}) = \mathfrak{b}_j$. Write $\rho^0(\mathfrak{a}) = \mathfrak{a}$ and $\rho^n(\mathfrak{a}) = \rho(\rho^{n-1}(\mathfrak{a})) = \rho^{-1}(\rho^{n+1}(\mathfrak{a}))$ for $n \in \mathbb{Z}$.

3 (f,p) Representations and their Arithmetic

In order to perform arithmetic with principal ideals, we represent a (generally unknown) primitive ideal $\mathfrak{a}$ by a known ideal $\mathfrak{b} = (\theta)\mathfrak{a}$ (where (θ) is the fractional principal ideal generated by $\theta \in K$) and an approximation of θ to a given precision and accuracy. More precisely:

Definition 3.1 *Let $p \in \mathbb{N}$ and $f \in \mathbb{R}$ with $1 \leq f < 2^p$. An* (f,p) representation *of a primitive ideal $\mathfrak{a}$ is a pair $(\mathfrak{b}, d)$ where*

1. *$\mathfrak{b}$ is an ideal equivalent to $\mathfrak{a}$,*
2. *$d, p \in \mathbb{Z}$ with $d > 0$ and $p \geq 0$,*
3. *if $\theta \in K$ with $\mathfrak{b} = (\theta)\mathfrak{a}$, then $\left|\dfrac{2^p\theta}{d} - 1\right| < \dfrac{f}{2^p}$.*

An (f,p) representation $(\mathfrak{b}, d)$ of $\mathfrak{a}$ is reduced *if $\mathfrak{b}$ is a reduced ideal.*

Informally speaking, $d2^{-p}$ is an approximation of the relative generator θ of $\mathfrak{b}$ with respect to $\mathfrak{a}$ to accuracy $f2^{-p}$, and p is the precision of the approximation. Note that for any primitive ideal $\mathfrak{a}$ and any $p \in \mathbb{N}$, $(\mathfrak{a}, 2^p + 1)$ is always a $(1,p)$ representation, and hence an (f,p) representation for any $f \in [1, 2^p)$, of $\mathfrak{a}$.

We now investigate the arithmetic of (f,p) representations.

Algorithm *MULT*

Input: *(f_i, p) representations $(\mathfrak{b}_i, d_i)$ of primitive ideals $\mathfrak{a}_i$ where $\mathfrak{b}_i = (Q_i, P_i)$ for $i = 1, 2$.*

Output: *An (f,p) representation $((S)\mathfrak{b}, d)$ of $\mathfrak{a}_1\mathfrak{a}_2$ where $\mathfrak{b} = (Q, P)$ is primitive and $f = 1 + f_1 + f_2 + f_1 f_2 2^{-p}$.*

Algorithm:

1. *Compute $G = \gcd\left(\dfrac{Q_1}{\sigma}, \dfrac{Q_2}{\sigma}\right)$ and solve $\dfrac{Q_1}{\sigma}X \equiv G \left(\mathrm{mod}\ \dfrac{Q_2}{\sigma}\right)$ for $X \in \mathbb{Z}$.*
2. *Compute $S = \gcd\left(\dfrac{P_1 + P_2}{\sigma}, G\right)$ and solve $Y\dfrac{P_1 + P_2}{\sigma} + ZG = S$ for $Y, Z \in \mathbb{Z}$.*
3. *Set $U \equiv XY(P_2 - P_1) + Z\dfrac{D - P_1^2}{Q_1} \left(\mathrm{mod}\ \dfrac{Q_2}{S}\right)$*
4. *Set $Q = \dfrac{Q_1 Q_2}{\sigma S^2}$, $P \equiv P_1 + U\dfrac{Q_1}{\sigma S} \pmod{Q}$, $d = \left\lceil \dfrac{d_1 d_2}{2^p} \right\rceil$.*

Write $((S)\mathfrak{b}, d) = \mathrm{MULT}(\mathfrak{b}_1, d_1, \mathfrak{b}_2, d_2)$ for the output of Algorithm MULT.

Lemma 3.2 *If $d_1, d_2 > 2^p$, then $d > 2^p$, and Algorithm MULT produces the correct output using $O(\log D)$ integer operations.*

Proof: The expressions for Q and P are simply given by the well-known ideal multiplication formulas, and the complexity result is clear. Let $\mathfrak{b}_i = (\theta_i)\mathfrak{a}_i$ for $i = 1, 2$. Then $\mathfrak{b} = (\theta_1\theta_2)\mathfrak{a}_1\mathfrak{a}_2$. Set $f_0 = f_1 + f_2 + f_1 f_2 2^{-p} = f - 1$. Then

$$1 - \frac{f_0}{2^p} < \frac{2^{2p}\theta_1\theta_2}{d_1 d_2} < 1 + \frac{f_0}{2^p}.$$

It is clear that $d > 2^p$ and $2^p\theta_1\theta_2/d < 1 + f2^{-p}$. If $\eta = d - d_1 d_2 2^{-p}$, then

$$2^p\theta_1\theta_2 > (d - \eta)\left(1 - \frac{f_0}{2^p}\right) = d\left(1 - \frac{f_0 + 2^p\eta d^{-1}}{2^p}\right) + \eta\frac{f_0}{2^p} > d\left(1 - \frac{f_0 + 1}{2^p}\right).$$

□

For the following algorithm, we assume that $s \in \mathbb{N}$ is sufficiently large so that

$$\frac{d_1}{2^s} < \frac{1}{10} \ .$$

Algorithm *REDUCE*

Input: *An (f, p) representation $((S)\mathfrak{b}_1, d_1)$ of some ideal $\mathfrak{a}$ where $\mathfrak{b}_1 = (Q_0, P_0)$ is primitive.*

Output: *A reduced $(f + 2, p)$ representation $(\mathfrak{b}, d)$ of $\mathfrak{a}$ where $\mathfrak{b} = (Q, P)$.*

Algorithm:

1. *Set $T_{-2} = -P_0 2^s + \lfloor 2^s\sqrt{D}\rfloor$, $T_{-1} = 2^s Q_0$, $M = \lfloor SQ_0 2^{p+s}/d_1\rfloor$, $j = 0$.*
2. *While $Q_{j-1} < 0$ or $P_{j-1} > \sqrt{D}$ do*
   (a) *Increase j by 1;*
   (b) *Set $q_{j-1} = \left\lfloor \dfrac{P_{j-1} + \sqrt{D}}{Q_{j-1}} \right\rfloor$, $P_j = q_{j-1}Q_{j-1} - P_{j-1}$, $Q_j = \dfrac{D - P_j^2}{Q_{j-1}}$;*
   (c) *Set $T_{j-1} = q_{j-1}T_{j-2} + T_{j-3}$;*
3. *If $T_{j-2} \le M$, then*
   (a) *Repeat*
      i. *Increase j by 1;*
      ii. *Set $q_{j-1} = \left\lfloor \dfrac{P_{j-1} + \sqrt{D}}{Q_{j-1}} \right\rfloor$, $P_j = q_{j-1}Q_{j-1} - P_{j-1}$, $Q_j = \dfrac{D - P_j^2}{Q_{j-1}}$;*

iii. Set $T_{j-1} = q_{j-1}T_{j-2} + T_{j-3}$.

until $T_{j-2} > M$.

(b) Set $\mathfrak{b} = (Q_{j-1}, P_{j-1})$, $d = \left\lceil \dfrac{d_1 T_{j-2}}{SQ_0 2^s} \right\rceil$.

Else

(c) Set $Q_0^* = Q_{j-1}$, $P_0^* = P_{j-1}$, $T_{-1}^* = T_{j-2}$, $T_0^* = T_{j-3}$, $j = 0$.

(d) Repeat

i. Increase j *by 1;*

ii. Set $Q_j^* = \dfrac{D - (P_{j-1}^*)^2}{Q_{j-1}^*}$, $q_j^* = \left\lfloor \dfrac{P_{j-1}^* + \sqrt{D}}{Q_j^*} \right\rfloor$, $P_j^* = q_j^* Q_j^* - P_{j-1}^*$;

iii. Set $T_j^* = T_{j-2}^* - q_j^* T_{j-1}^*$;

until $T_{j-1}^* \leq M$.

(e) Set $\mathfrak{b} = (Q_{j-1}^*, P_{j-1}^*)$, $d = \left\lceil \dfrac{d_1 T_{j-2}^*}{SQ_0 2^s} \right\rceil$.

Write $(\mathfrak{b}, d) = \text{REDUCE}((S)\mathfrak{b}_1, d_1)$ for the output of Algorithm REDUCE. With the notation of the algorithm, we let

$$\mathfrak{b}_j = (Q_{j-1}, P_{j-1}),\ d_j = \left\lceil \frac{d_1 T_{j-2}}{SQ_0 2^s} \right\rceil,\ \mathfrak{b}_j^* = (Q_{j-1}^*, P_{j-1}^*),\ d_j^* = \left\lceil \frac{d_1 T_{j-2}^*}{SQ_0 2^s} \right\rceil$$

for $j \in \mathbb{N}$. We first show that all the loops in the above algorithm terminate.

Lemma 3.3 *Let s be as described above and assume that $d_1 > 2^p$. Then the loops in steps 2, 3 (a), and 3 (d) in Algorithm REDUCE terminate.*

Proof: The while loop in step 2 terminates after $O(\log(Q_0/D))$ iterations, once the first reduced ideal $\mathfrak{b}_l$ ($l \in \mathbb{N}$) is encountered. Then $\Psi_l \leq 1$. We consider two cases.

Case 1: $T_{l-2} \leq M$. Then it suffices to show the existence of an index $k > l$ with $T_{k-2} > M$.

For $i \in \mathbb{N}$, we have $\mathfrak{b}_i = (\Psi_i)\mathfrak{b}_1$ where Ψ_i is given by (2.2). Write $\Psi_i = (G_{i-2} + B_{i-2}\sqrt{D})/Q_0$ with $G_{i-2}, B_{i-2} \in \mathbb{Z}$ and $B_{i-2} \geq 0$. Set

$$\hat{\Psi}_i = \frac{(G_{i-2} + B_{i-2} 2^{-s} \lfloor 2^s \sqrt{D} \rfloor}{Q_0}.$$

Then

$$0 < \Psi_i - \hat{\Psi}_i < \frac{B_{i-2}}{Q_0 2^s}$$

and $T_{i-2} = 2^s Q_0 \hat{\Psi}_i$ for $i \in \mathbb{N}$. It is easy to see that $T_{i-2} > M$ if and only if $\hat{\Psi}_i > S2^p/d_1$.

Since $\Psi_1 = 1$ and the sequence $(\Psi_i)_{i \geq l}$ is increasing, there exists an index $k > l$ for which $\Psi_{k-1} \leq 3S/2 < \Psi_k$. We have $\Psi_k = \psi_{k-1}\Psi_{k-1}$ with $\psi_{k-1} = (P_{k-1} + \sqrt{D})/Q_{k-2} < 2\sqrt{D}$, hence $\Psi_k < 3S\sqrt{D}$. Together with $|\overline{\Psi}_k| < 1$, obtain

$$0 < \frac{B_{k-2}}{Q_0 2^s} \leq \frac{\Psi_k + |\overline{\Psi}_k|}{2^{s+1}\sqrt{D}} < \frac{3S\sqrt{D}+1}{2^{s+1}\sqrt{D}} < S \cdot 2^{1-s} < \frac{S}{2} \tag{3.6}$$

and hence $\hat{\Psi}_k > \Psi_k - B_{k-2}/Q_0 2^s > S > S2^p/d_1$. Then $T_{k-2} > 2^{p+s} Q_0 S/d_1$ and hence $T_{k-2} > M$.

Case 2: $T_{l-2} > M$. Then $T^*_{-1} > M$ and we need to establish the existence of an index $k \in \mathbb{N}$ with $T^*_k \leq M$. We have $\mathfrak{b}^*_i = (\Psi^*_i)\mathfrak{b}^*_1$ where the Ψ^*_i are given by (2.5), and as before, we define an approximation $\hat{\Psi}^*_i$ to Ψ^*_i ($i \in \mathbb{N}$) for which

$$0 < \operatorname{sgn}(B_{i-2})(\Psi^*_i - \hat{\Psi}^*_i) < \frac{|B^*_{i-2}|}{Q_0 2^s}$$

and $T^*_{i-2} = 2^s Q_0 \hat{\Psi}^*_i$ for $i \in \mathbb{N}$. Once again, $T^*_{i-2} > M$ if and only if $\hat{\Psi}^*_i > S2^p/d_1$.

Now $B^*_{-1} > 0$ implies that $\Psi^*_1 > \hat{\Psi}^*_1 > S2^p/d_1$. Therefore, there must exist an index $k \in \mathbb{N}$ with $\Psi^*_k > S2^p/d_1 \geq \Psi^*_{k+1}$. If $T^*_{k-1} \leq M$, there is nothing to prove, so assume $T^*_{k-1} > M$. Then $\hat{\Psi}^*_{k+1} > S2^p/d_1 > \Psi^*_{k+1}$ and hence $B^*_{k-1} < 0$. The identity $\Psi^*_{k+2} = \psi^*_{k+1}\Psi^*_{k+1}$ implies

$$B^*_k = \frac{G^*_{k-1} - B^*_{k-1}P^*_k}{Q^*_k} = \frac{Q_0\Psi^*_{k+1} - B^*_{k-1}(P^*_k + \sqrt{D})}{Q^*_k} > 0.$$

Therefore $\hat{\Psi}^*_{k+2} < \Psi^*_{k+2} < \Psi^*_{k+1} \leq S2^p/d_1$ and hence $T^*_k \leq M$. □

Lemma 3.4 *Let $((S)\mathfrak{b}_1, d_1)$ be the input of Algorithm REDUCE and let s be as described above. Assume that $2^p < d_1 < 9S^2 Q_0 2^{p-2}$.*

1. *Let $j \in \mathbb{N}$ be defined by the last iteration of the loop in step 3 (a) of Algorithm REDUCE, so $\mathfrak{b}_j$ is reduced and $T_{j-3} \leq M < T_{j-2}$. Then $d_{j-1} \leq 2^p < d_j$, and if $P_{j-1} \neq \lfloor\sqrt{D}\rfloor$, then $d_j \leq 3Q_{j-1}2^{p-1}$.*
2. *Let $j \in N$ be defined by the last iteration of the loop in step 3 (d) of Algorithm REDUCE, so $\mathfrak{b}^*_j$ is reduced and $T^*_{j-1} \leq M < T^*_{j-2}$. Then $d^*_{j+1} \leq 2^p < d^*_j$, and if $P^*_{j-1} \neq \lfloor\sqrt{D}\rfloor$, then $d^*_j \leq 3Q^*_{j-1}2^{p-1}$.*

Proof: We use the same notation as in the proof of the previous lemma. It is easy to see that $T_{i-2} > M$ if and only if $d_i > 2^p$, so $d_{j-1} \leq 2^p < d_j$; similarly, $T^*_{i-2} > M$ if and only if $d^*_i > 2^p$, so $d_{j+1} \leq 2^p < d_j$. Furthermore, $d_1 2^{-s} < 1/4$ and $2^p + 1/2 < 3 \cdot 2^{p-1}$.

Once again, let $\mathfrak{b}_l$ be first reduced ideal computed in the algorithm and suppose first that $T_{l-2} \leq M$. Let k be as in case 1 of the proof of Lemma 3.3. Then $j \leq k$ and $B_{k-2}/Q_0 < 2S$ by (3.6). Since B_i strictly increases with i, it follows that

$$\frac{d_1}{S}\Psi_{j-1} < \frac{d_1}{S}\left(\hat{\Psi}_{j-1} + \frac{B_{j-3}}{Q_0 2^s}\right) < d_{j-1} + \frac{d_1}{2^s}\frac{B_{k-2}}{SQ_0} < 2^p + 1/2 < 3 \cdot 2^{p-1}.$$

If $P_{j-1} \neq \lfloor \sqrt{D} \rfloor$, then $\psi_{j-1} = Q_{j-1}/(\sqrt{D} - P_{j-1}) < Q_{j-1}$, hence

$$\frac{d_1}{S}\hat{\Psi}_j \leq \frac{d_1}{S}\Psi_j = \frac{d_1}{S}\psi_{j-1}\Psi_{j-1} < 3Q_{j-1}2^{p-1},$$

and we have $d_j \leq 3Q_{j-1}2^{p-1}$.

Now suppose that $T_{l-2} > M$ and let k be as in case 2 of the proof of Lemma 3.3. Then $j \leq k+1$. Let $i \leq k$, so $\Psi_i^* > S2^p/d_1$. Then

$$\begin{aligned} |\overline{\Psi_i^*}| &= \frac{Q_{i-1}^*}{\Psi_i^* Q_0} < \frac{2\sqrt{D}d_1}{SQ_0 2^p} < \frac{9}{2}S\sqrt{D}, \\ |\overline{\Psi_{k+1}^*}| &= |\overline{\psi_k^*}||\overline{\Psi_k^*}| < \frac{P_{k-1}^* + \sqrt{D}}{Q_{k-1}^*}\frac{d_1 Q_{k-1}^*}{SQ_0 2^p} < \frac{2\sqrt{D}d_1}{SQ_0 2^p} < \frac{9}{2}S\sqrt{D}. \end{aligned}$$

Together with $\Psi_i^*, \Psi_{k+1}^* < 1$, this implies for all $i \leq k+1$,

$$\frac{d_1|B_{i-2}^*|}{SQ_0 2^s} \leq \frac{d_1}{S2^s}\frac{|\Psi_i^*| + |\overline{\Psi_i^*}|}{2\sqrt{D}} < \frac{d_1}{2^s}\left(\frac{1}{2S\sqrt{D}} + \frac{9}{4}\right) < \frac{1}{4}.$$

If $j \leq k$, then

$$\begin{aligned} \frac{d_1}{S}\hat{\Psi}_j^* &< \frac{d_1}{S}\left(\frac{1}{\psi_j^*}\left(\hat{\Psi}_{j+1}^* + \frac{|B_{j-1}^*|}{Q_0 2^s}\right) + \frac{|B_{j-2}^*|}{Q_0 2^s}\right) \\ &< \frac{1}{\psi_j^*}\left(d_{j+1}^* + \frac{1}{4} + \frac{1}{4}\psi_j^*\right) < \frac{3\cdot 2^{p-1}}{\psi_j^*}, \end{aligned}$$

where we use $\psi_j^* < 1$ for the last inequality. Since $\psi_j^* = (\sqrt{D} - P_{j-1}^*)/Q_{j-1}^* > 1/Q_{j-1}^*$ if $P_{j-1}^* \neq \lfloor \sqrt{D} \rfloor$, we obtain $d_j^* \leq 3Q_{j-1}^* 2^{p-1}$.

If $j = k+1$, then $d_1\Psi_j^*/S < 2^p$, so $d_1\hat{\Psi}_j^*/S < 2^p + 1/4 < 3\cdot 2^{p-1}$ and hence $d_j^* < 3\cdot 2^{p-1}$. □

Corollary 3.5 *Assume the conditions and notation of Lemma 3.4.*

1. $d_i - 2 < d_1\Psi_i/S < d_i + 1$ *for* $i \in \{j, j-1\}$.
2. $d_i^* - 2 < d_1\Psi_i^*/S < d_i^* + 1$ *for* $i \in \{j, j+1\}$.

Proof: If k is defined as in case 1 of the proof of Lemma 3.3, we have by (3.6), $0 < d_1(\Psi_i - \hat{\Psi}_i)/S < d_1 2^{1-s} < 1$ for all $i \leq k$. If k is as in case 2 of the same proof, then again $d_1|\Psi_i^* - \hat{\Psi}_i^*|/S < 1/4$ for all $i \leq k+1$. Also, if $j = k+1$, then $d_1\Psi_j^*/S > 2^p - 1/4 > 2^p - 1$, so $|\overline{\Psi_j^*}| < d_1 Q_{j-1}^*/SQ_0(2^p - 1)$ and

$$|\overline{\Psi_{j+1}^*}| = |\overline{\psi_j^*}||\overline{\Psi_j^*}| < \frac{2\sqrt{D}d_1}{SQ_0(2^p - 1)} < \frac{18\sqrt{D}S2^{p-2}}{2^p - 1}.$$

Together with $\Psi^*_{j+1} < 1$, we obtain

$$\begin{aligned}\frac{d_1}{S}|\Psi^*_{j+1} - \hat{\Psi}^*_{j+1}| &< \frac{d_1}{S}\frac{B^*_{j-1}}{Q_0 2^s} \le \frac{d_1}{S2^s}\frac{\Psi^*_{j+1} + |\overline{\Psi^*_{j+1}}|}{2\sqrt{D}} < \frac{d_1}{2^s}\left(\frac{1}{2S\sqrt{D}} + \frac{9\cdot 2^{p-2}}{2^p - 1}\right) \\ &= \frac{d_1}{2^s}\left(\frac{1}{2S\sqrt{D}} + \frac{9}{4}\right) + \frac{d_1}{2^s}\frac{9/4}{2^p - 1} < \frac{1}{2},\end{aligned}$$

where we use the fact that $d_1 2^{-s} < 1/10$ for the last inequality. Now is easy to see that for any two positive real numbers α and $\hat{\alpha}$, the inequality $|\alpha - \hat{\alpha}| < 1$ implies $\lceil\hat{\alpha}\rceil - 2 < \alpha < \lceil\hat{\alpha}\rceil + 1$. □

Theorem 3.6 *Assuming the conditions of Lemma 3.4 and the notation of Algorithm REDUCE, the following hold.*

1. $(\mathfrak{b}, d)$ *is a reduced* $(f+2, p)$ *representation of* $\mathfrak{a}$.
2. *The loops in step 3 in the algorithm are executed at most* $O(\log(SD))$ *and* $O(\log D)$ *times, respectively.*
3. $T_i, T_i^* = O(SQ_0\sqrt{D}2^s)$ *throughout the algorithm.*

Proof: Let $(S)\mathfrak{b}_1 = (\theta)\mathfrak{a}$, $\mathfrak{b} = (\Psi)\mathfrak{b}_1$, $d = \lceil d_1\hat{\Psi}/S\rceil$.

1. From $f < 2^p$ and $d_1(1 - f2^{-p}) < 2^p\theta < d_1(1 + f2^{-p})$, we obtain from the previous Corollary,

$$\begin{aligned}\frac{2^p\Psi\theta}{S} &< \frac{\Psi}{S}d_1\left(1 + \frac{f}{2^p}\right) \le (d+1)\left(1 + \frac{f}{2^p}\right) \\ &< d\left(1 + \frac{f}{2^p} + \frac{2}{d}\right) < d\left(1 + \frac{f+2}{2^p}\right)\end{aligned}$$

and

$$\begin{aligned}\frac{2^p\Psi\theta}{S} &> \frac{\Psi}{S}d_1\left(1 - \frac{f}{2^p}\right) \ge (d-2)\left(1 - \frac{f}{2^p}\right) \\ &> d\left(1 - \frac{f}{2^p} - \frac{2}{d}\right) > d\left(1 - \frac{f+2}{2^p}\right).\end{aligned}$$

2. Suppose step 3 (a) is executed j times. Then by (2.3) $\Psi_j > 2^{\lfloor(j-1)/2\rfloor}$, and by Corollary 3.5 and Lemma 3.4,

$$\Psi_j < \frac{S}{d_1}(d_j + 1) < \frac{S}{2^p}(3Q_{j-1}2^{p-1} + 1) = O(S\sqrt{D})$$

because $\mathfrak{b}_j$ is reduced.

Now let j be the number of iterations of step 3 (d). Then again by (2.3) $\Psi^*_j < 2^{-\lfloor(j-1)/2\rfloor}\Psi^*_1$ where $d_1\Psi^*_1/S = O(\sqrt{D}2^p)$. It follows that

$$2^p - 2 < d^*_j - 2 \le \frac{d_1\Psi^*_j}{S} < 2^{-\lfloor(j-1)/2\rfloor}O(\sqrt{D}2^p),$$

so $2^{\lfloor(j-1)/2\rfloor} = O(\sqrt{D})$.

3. Let j be as in part 1 of Lemma 3.4. Since $d_1 > 2^p$ and $d_j \le 3Q_{j-1}2^{p-1} < 3\sqrt{D}2^p$, we see that $\hat{\Psi}_j \le Sd_j/d_1 < 3\sqrt{D}S$. Using the bound $B_{j-2} < 2Q_0S$ from (3.6), we obtain for $1 \le i \le j$:

$$T_{i-2} = 2^s Q_0 \hat{\Psi}_i \le 2^s Q_0 \Psi_i \le 2^s Q_0 \Psi_j < 2^s Q_0 \hat{\Psi}_j + B_{j-2} = O(2^s \sqrt{D} S Q_0).$$

Now let j be as in part 2 of Lemma 3.4. Since $\Psi_i^* \le \Psi_1^* = O(S\sqrt{D})$ and $|\overline{\Psi_i^*}| < 1$ for $1 \le i \le j$, we obtain

$$T_{i-2}^* = 2^s Q_0 \hat{\Psi}_i^* \le 2^s Q_0 \Psi_i^* + |B_{i-2}^*| \le 2^s Q_0 \Psi_i^* + \frac{Q_0}{2\sqrt{D}}(\Psi_i^* + |\overline{\Psi_i^*}|) = O(2^s \sqrt{D} Q_0 S).$$

□

We combine the previous two algorithms as follows:

Algorithm *MR*

Input: *Reduced* (f_i, p) *representations* $(\mathfrak{b}_i, d_i)$ *of primitive ideals* $\mathfrak{a}_i$ $(i = 1, 2)$.

Output: *A reduced* (f, p) *representation* $(\mathfrak{b}, d)$ *of* $\mathfrak{a}_1\mathfrak{a}_2$ *where* $f = 3 + f_1 + f_2 + f_1 f_2 2^{-p}$.

Algorithm:

1. $((S)\mathfrak{c}, e) = MULT(\mathfrak{b}_1, d_1, \mathfrak{b}_2, d_2)$.
2. $(\mathfrak{b}, d) = REDUCE((S)\mathfrak{c}, e)$.

Write $(\mathfrak{b}, d) = \mathrm{MR}(\mathfrak{b}_1, d_1, \mathfrak{b}_2, d_2)$.

Theorem 3.7 *Let* $(\mathfrak{b}_1, d_1), (\mathfrak{b}_2, d_2)$ *be the inputs and* $(\mathfrak{b}, d)$ *the output of Algorithm MR. Write* $\mathfrak{b}_1 = (Q_1, P_1)$, $\mathfrak{b}_2 = (Q_2, P_2)$, $\mathfrak{b} = (Q, P)$, *and suppose that* $2^p < d_i < 3Q_i 2^p$ *for* $i = 1, 2$. *Then* $(\mathfrak{b}, d)$ *is a reduced* (f, p) *representation of* $\mathfrak{a}_1\mathfrak{a}_2$ *with* $2^p < d < 3Q2^p$. *Also, Algorithm MR requires* $O(\log D)$ *integer operations to compute* $(\mathfrak{b}, d)$, *and the largest value computed in the algorithm is bounded by* $O(D^{3/2}2^p)$.

Proof: We have $(S)\mathfrak{c} = \mathfrak{b}_1\mathfrak{b}_2$. Let $\mathfrak{c} = (\tilde{Q}, \tilde{P})$. Then $((S)\mathfrak{c}, e)$ is a (g, p) representation of $\mathfrak{a}_1\mathfrak{a}_2$ where $g = 1 + f_1 + f_2 + f_1 f_2 2^{-p}$ and $2^p < e \le d_1 d_2 2^{-p} < 9S^2\tilde{Q}2^{p-2}$. Also, $(\mathfrak{b}, d)$ is an (f, p) representation of $\mathfrak{a}_1\mathfrak{a}_2$ with $2^p < d < 3Q2^{p-1}$ and $f = g + 2$. The complexity result is clear since $S \le \min\{Q_1, Q_2\} < 2\sqrt{D}$, and the largest value computed is $O(\tilde{Q}S\sqrt{D}2^p) = O(D^{3/2}2^p)$. □

We point out that the upper bound $3Q2^{p-1}$ on d required $P < \lfloor\sqrt{D}\rfloor$. In the very unlikely case where equality occurs, we can only achieve $d < 3aQ_{j-1}2^{p-1}$ where $a = (\sqrt{D} - \lfloor\sqrt{D}\rfloor)^{-1} > 1$. However, letting $\mathfrak{b} = (\Psi)\mathfrak{c}$, one can still achieve $e|\Psi - \hat{\Psi}|/S < 1$ at the expense of higher precision s' in Algorithm REDUCE. Here, s' must be chosen so that

$$\frac{e}{2^{s'}} < \frac{1}{10} \ .$$

Let T_j ($j \in \mathbb{N}$) be the sequence of values computed in steps 2 and 3 of Algorithm REDUCE with respect to precision s, and let T'_j be the analogous sequence if precision s' is used. Then one can recover T'_{j-1} and T'_j from T_{j-1} and T_j using the formulas

$$\begin{aligned}(2^{2s}D - \delta^2)T'_{j-1} &= (2^{s+s'}D - \delta\delta')T_{j-1} + (2^s\delta' - s^{s'}\delta)(Q_jT_j - P_{j+1}T_{j-1}),\\ (2^{2s}D - \delta^2)T'_j &= (2^{s+s'}D - \delta\delta')T_j + (2^s\delta' - s^{s'}\delta)(P_{j+1}T_j + Q_{j+1}T_{j-1}),\end{aligned}$$

where $\delta = \lfloor\sqrt{D}2^s\rfloor$ and $\delta' = \lfloor\sqrt{D}2^{s'}\rfloor$. Analogous formulas can be derived for T^*_{j-1} and T^*_j. In any case, it is always possible to achieve $a < 2$, for example, if we choose D to be of the form $D = A^2 + R$ with $A, R \in \mathbb{N}$ and $A + 1 \leq R \leq 2A$.

We conclude this section with a method for exponentiation with (f, p) representations. This algorithm is an adaptation of the standard binary exponentiation technique.

Algorithm *EXP*

Input: *$n \in \mathbb{N}$ and a reduced (f_0, p) representation $(\mathfrak{b}_0, d_0)$ of some ideal $\mathfrak{a}$.*

Output: *A reduced (f, p) representation $(\mathfrak{b}, d)$ of $\mathfrak{a}^n$ for suitable $f \in [1, 2^p)$.*

Algorithm:

1. *Compute the binary representation of n, say $n = \sum_{i=0}^{k} b_i 2^{k-i}$ ($b_0 = 1$, $b_i \in \{0, 1\}$ for $1 \leq i \leq k$, $k = \lfloor \log_2 n \rfloor$).*
2. *Set $(\mathfrak{c}, e) = (\mathfrak{b}_0, d_0)$.*
3. *For $i = 1$ to k do*
   - *(a) $(\mathfrak{b}, d) = MR(\mathfrak{c}, e, \mathfrak{c}, e)$.*
   - *(b) If $b_i = 1$ then replace $(\mathfrak{b}, d)$ by $MR(\mathfrak{b}, d, \mathfrak{b}_0, d_0)$.*
   - *(c) $(\mathfrak{c}, e) = (\mathfrak{b}, d)$.*

Write $(\mathfrak{b}, d) = \mathrm{EXP}(\mathfrak{b}_0, d_0, n)$. Before we analyze Algorithm EXP, we require an auxiliary lemma.

Lemma 3.8 *Let $p, k \in \mathbb{N}$ and $h \in \mathbb{R}$ with $p \geq 8$ and $h \geq \max\{16, k\}$. Define a sequence $(a_i)_{i\geq 0}$ recursively via*

$$a_0 \in \mathbb{R}_+, \quad a_i = 6 + \left(\left(1 + \frac{1}{h}\right)^2 + \frac{3}{2^p}\right)a_0 + \left(2 + \frac{1}{h}\right)a_{i-1} \quad (i \in \mathbb{N}).$$

Then $a_k < (3.43a_0 + 9.9)2^k$.

Proof: For brevity, set $g = 2 + h^{-1}$. Then it is easy to verify that the closed form for $(a_i)_{i\geq 0}$ is

$$a_i = \left(g^{i+1} - g + 1 + \frac{g^i - 1}{g-1} \cdot \frac{3}{2^p}\right) a_0 + 6\,\frac{g^i - 1}{g-1} \quad (i \in \mathbb{N}).$$

Now $(g^i - 1)/(g-1) < g^i$ and the multiple of a_0 in the above equality is bounded above by $g^{i+1} + 3 \cdot 2^{-8} g^i$. Using $h \geq 16$ and $p \geq 8$, we obtain

$$\begin{aligned} a_i &< g^i \left(\left(2 + \frac{1}{16} + \frac{3}{256}\right) a_0 + 6\right) < g^i (2.08 a_0 + 6) \\ &= 2^i \left(1 + \frac{1}{2h}\right)^i (2.08 a_0 + 6) < 2^i \exp\left(\frac{i}{2h}\right) (2.08 a_0 + 6)\ . \end{aligned}$$

Since $h \geq k$, we have $a_k \leq 2^k \exp(0.5)(2.08 a_0 + 6) < (3.43 a_0 + 9.9) 2^k$. □

Theorem 3.9 *Let $(\mathfrak{b}_0, d_0)$ and n be the inputs and $(\mathfrak{b}, d)$ the output of Algorithm EXP. Write $\mathfrak{b}_0 = (\tilde{Q}, \tilde{P})$, $\mathfrak{b} = (Q, P)$, and suppose that $d_0 < 9 S^2 \tilde{Q} 2^{p-2}$.*

1. *$\mathfrak{b}$ is a reduced (f, p) representation of $\mathfrak{b}_0$ with $2^p < d < 3Q2^{p-1}$.*
2. *Suppose $p \geq 8$ and let $h \in \mathbb{R}_+$ with $h \geq \max\{16, \log_2 n\}$. Set $e = 3.43 f_0 + 9.9$. If $hen < 2^p$, then $f < en$ and hence $hf < 2^p$.*

Also, Algorithm EXP requires $O(\log n \log D)$ integer operations.

Proof: After the i-th iteration of step 3 ($1 \leq i \leq k$), let $\mathfrak{b} = \mathfrak{b}_i$ and $d = d_i$. If we set $s_0 = b_0 = 1$ and $s_i = 2s_{i-1} + b_i$ for $1 \leq i \leq k$, then $(\mathfrak{b}_i, d_i)$ is an (f_i, p) representation of $\mathfrak{a}^{s_i}$ where

$$f_i = 3 + f_0 + \left(3 + 2f_{i-1} + \frac{f_{i-1}^2}{2^p}\right) + f_0 \left(3 + 2f_{i-1} + \frac{f_{i-1}^2}{2^p}\right) 2^{-p}$$

for $1 \leq i \leq k$. Set $f = f_k$. Since $s_k = n$, Algorithm EXP produces a reduced (f, p) representation of $\mathfrak{a}^n$ with $2^p < d < 3Q2^{p-1}$.

Let $(a_i)_{i\geq 0}$ be as in Lemma 3.8 with $a_0 = f_0$. Since $h \geq k$, we have $a_k < e2^k \leq en$, so $ha_i < 2^p$ for $0 \leq i \leq k$. Then $hf_0 < 2^p$ and inductively, $f_i < a_i$, so $f_i < en$ and $hf_i < 2^p$; in particular, $f < en$ and $hf < 2^p$. □

4 The Protocol

Our goal is to establish a protocol by which both parties are able to compute a unique reduced principal ideal $\mathfrak{k}$. This ideal will be an (f, p) representation of $\mathfrak{r}^{ab}$ for suitable f where $\mathfrak{r}$ is a public reduced starting ideal and a and b are the respective secret exponents. Since the two communicants

follow different exponentiation procedures for computing a candidate for $\mathfrak{k}$, they may not arrive at the same ideal. However, we will see that their respective candidates lie within only five reduction steps (2.1) of each other. By exchanging five more bits of information, both parties will be able to agree on a unique key ideal.

Lemma 4.1 *Let $\mathfrak{r}$ be a reduced principal ideal and let $p, a, b, B \in \mathbb{Z}$ with $0 < a, b \leq B$, $B \geq 36$, and $2^p \geq 46B^2 \max\{16, \log_2 B\}$. If $(\mathfrak{k}, d) = EXP(EXP(\mathfrak{r}, 2^p + 1, a), b)$, then $\mathfrak{k}$ is an (f, p) representation of $\mathfrak{r}^{ab}$ with $16f < 2^p$.*

Proof: Clearly, $(\mathfrak{r}, 2^p + 1)$ is a reduced $(1, p)$ representation of $\mathfrak{r}$. Set $h = \max\{16, \log_2 B\}$, then $h \geq \max\{16, \log_2 a, \log_2 b\}$. Let $(\mathfrak{a}, e) = \mathrm{EXP}(\mathfrak{r}, 2^p + 1, a)$. Since $h(3.43 \cdot 1 + 9.9)B < 2^p$, by Theorem 3.9, $(\mathfrak{a}, e)$ is a reduced (g, p) representation of $\mathfrak{r}^a$ with $g < 13.33a \leq 13.33B$ and $hg < 2^p$. Similarly, $h(3.43g + 9.9)b < h(45.72B^2 + 9.9B) < 46B^2 \max\{16, \log_2 B\} < 2^p$. Since $h \geq 16$, $(\mathfrak{k}, d)$ is a reduced (f, p) representation of $\mathfrak{r}^{ab}$ with $16f < 2^p$. □

Theorem 4.2 *Let $(\mathfrak{b}_1, d_1)$ and $(\mathfrak{b}_2, d_2)$ be two (f, p) representations of some ideal $\mathfrak{a}$ with $p \geq 3$ and $7f \leq 2^p$. Let $\mathfrak{k} = (\kappa)\mathfrak{b}_1$, $\rho^{-1}(\mathfrak{k}) = (\kappa_-)\mathfrak{b}_1$, $\mathfrak{m} = (\mu)\mathfrak{b}_2$, $\rho^{-1}(\mathfrak{m}) = (\mu_-)\mathfrak{b}_2$ where $\mathfrak{k}$ and $\mathfrak{m}$ are reduced ideals with $d_1\kappa, d_2\mu > 2^p - 2$ and $d_1\kappa_-, d_2\mu_- < 2^p + 1$. Then $\mathfrak{k} \in \{\rho^{-2}(\mathfrak{m}), \rho^{-1}(\mathfrak{m}), \mathfrak{m}, \rho(\mathfrak{m}), \rho^2(\mathfrak{m})\}$.*

Proof: Suppose $\mathfrak{k} = \rho^i(\mathfrak{m})$ with $|i| \geq 3$ and assume without loss of generality that $i > 0$ (the case $i < 0$ can be proved by reversing the roles of $\mathfrak{k}$ and $\mathfrak{m}$). Then $\rho^{-1}(\mathfrak{k}) = (\alpha)\mathfrak{m}$ where $\alpha > 2$ by (2.3). Let $\mathfrak{b}_i = (\theta_i)\mathfrak{a}$ for $i = 1, 2$ and assume that $\theta_1, \theta_2, \kappa, \kappa_-, \mu, \mu_- > 0$. Then $\kappa_-\theta_1 > 2\mu\theta_2$. Now

$$\kappa_-\theta_1 < \kappa_- \frac{d_1}{2^p}\left(1 + \frac{f}{2^p}\right) < \frac{2^p + 1}{2^p}\left(1 + \frac{f}{2^p}\right),$$

$$\mu\theta_2 > \mu \frac{d_2}{2^p}\left(1 - \frac{f}{2^p}\right) > \frac{2^p - 2}{2^p}\left(1 - \frac{f}{2^p}\right),$$

so $(2^p + 1)(1 + f2^{-p}) > 2(2^p - 2)(1 - f2^{-p})$, implying

$$\frac{f}{2^p} > \frac{1}{3} \cdot \frac{2^p - 5}{2^p - 1} = \frac{1}{3}\left(1 - \frac{4}{2^p - 1}\right) > \frac{1}{3} \cdot \frac{3}{7} = \frac{1}{7},$$

a contradiction. □

Corollary 4.3 *Let $\mathfrak{r}$, p, B, a, b be as in Lemma 4.1 and let $(\mathfrak{k}, k) = EXP(EXP(\mathfrak{r}, 2^p + 1, a), b)$ and $(\mathfrak{m}, m) = EXP(EXP(\mathfrak{r}, 2^p + 1, b), a)$. Then $\mathfrak{k} \in \{\rho^{-2}(\mathfrak{m}), \rho^{-1}(\mathfrak{m}), \mathfrak{m}, \rho(\mathfrak{m}), \rho^2(\mathfrak{m})\}$.*

Proof: By Lemma 4.1, $(\mathfrak{k}, k)$ and $(\mathfrak{m}, m)$ are (f, p) representations of $\mathfrak{r}^{ab}$ with $16f < 2^p$.

Let $(\mathfrak{a}, e) = \mathrm{EXP}(\mathfrak{r}, 2^p + 1, a)$, and let $(\mathfrak{b}_1, d_1)$ be the quantity computed by Algorithm EXP on input $(\mathfrak{a}, e, b)$ before the very last reduction step. By Theorem 3.9 and Lemma 4.1, $(\mathfrak{b}_1, d_1)$ is a (g, p) representation of $\mathfrak{r}^{ab}$ with $16g < 16f < 2^p$. Once $(\mathfrak{b}_1, d_1)$ is found, Algorithm EXP generates $(\mathfrak{k}, k)$ and $(\rho^{-1}(\mathfrak{k}), k_-)$. If $\mathfrak{k} = (\kappa)\mathfrak{b}_1$ and $\rho^{-1}(\mathfrak{k}) = (\kappa_-)\mathfrak{b}_1$, then by Lemma 3.4, $k_- \leq 2^p < k$, and by

Corollary 3.5, $k_- - 2 < d_1\kappa_- < k_- + 1$ and $k - 2 < d_1\kappa < k + 1$. It follows that $d_1\kappa > 2^p - 2$ and $d_1\kappa_- < 2^p + 1$.

Similarly, let $(\mathfrak{b}, e') = \mathrm{EXP}(\mathfrak{r}, 2^p+1, b)$, and let $(\mathfrak{b}_2, d_2)$ be the quantity generated by algorithm EXP on input $(\mathfrak{b}, e', a)$ before the very last reduction step when computing $(\mathfrak{m}, m) = \mathrm{EXP}(\mathfrak{b}, e', a)$. Then $(\mathfrak{b}_2, d_2)$ is also a (g, p) representation of $\mathfrak{r}^{ab}$ with $16g < 2^p$, and if $\mathfrak{m} = (\mu)\mathfrak{b}_2$ and $\rho^{-1}(\mathfrak{m}) = (\mu_-)\mathfrak{b}_2$, then $d_2\mu > 2^p - 2$ and $d_2\mu_- < 2^p + 1$. Therefore, the conditions of Theorem 4.2 are satisfied and the Corollary follows. □

Lemma 4.4 *Assume the notation and conditions of Theorem 4.2 and suppose that $p \geq 9$ and $16f < 2^p$.*

1. *If $d_1\kappa_- < 7 \cdot 2^{p-3} + 1$, then $\mathfrak{k} \in \{\rho^{-2}(\mathfrak{m}), \rho^{-1}(\mathfrak{m}), \mathfrak{m}\}$.*
2. *If $d_1\kappa > 5 \cdot 2^{p-2} - 1$, then $\mathfrak{k} \in \{\rho^2(\mathfrak{m}), \rho(\mathfrak{m}), \mathfrak{m}\}$. If $d_2\mu > 5 \cdot 2^{p-2} - 1$, then $\mathfrak{k} \in \{\rho^{-2}(\mathfrak{m}), \rho^{-1}(\mathfrak{m}), \mathfrak{m}\}$.*
3. *If $d_1\kappa < 5 \cdot 2^{p-2} + 1$, then $\mathfrak{k} \neq \rho^2(\mathfrak{m})$.*
4. *If $d_1\kappa_- > 7 \cdot 2^{p-3} - 1$, then $\mathfrak{k} \neq \rho^{-2}(\mathfrak{m})$.*
5. *If $d_1\kappa_- > 7 \cdot 2^{p-3} - 1$ and $d_2\mu < 5 \cdot 2^{p-2} + 1$, then $\mathfrak{k} \neq \rho^{-1}(\mathfrak{m})$.*
6. *Suppose $d_1\kappa < 5 \cdot 2^{p-2} + 1$. If $d_2\mu < 7 \cdot 2^{p-2} + 1$, then $\mathfrak{k} \neq \rho^{-2}(\mathfrak{m})$, and if $d_2\mu > 7 \cdot 2^{p-2} - 1$, then $\mathfrak{k} \neq \mathfrak{m}$.*
   Suppose $d_2\mu < 5 \cdot 2^{p-2} + 1$. If $d_1\kappa < 7 \cdot 2^{p-2} + 1$, then $\mathfrak{k} \neq \rho^2(\mathfrak{m})$, and if $d_1\kappa > 7 \cdot 2^{p-2} - 1$, then $\mathfrak{k} \neq \mathfrak{m}$.

Proof: We only prove parts 1 and 5; the other parts follow similarly. Note also that the second statements in parts 2 and 6 are immediate corollaries of the respective first statements of these parts. By Theorem 4.2, $\mathfrak{k} \in \{\rho^{-2}(\mathfrak{m}), \rho^{-1}(\mathfrak{m}), \mathfrak{m}, \rho(\mathfrak{m}), \rho^2(\mathfrak{m})\}$.

For part 1, we have

$$\kappa_-\theta_1 < \frac{7 \cdot 2^{p-3} + 1}{2^p}\left(1 + \frac{f}{2^p}\right) < \left(\frac{7}{8} + \frac{1}{2^9}\right)\left(1 + \frac{1}{16}\right) = \frac{7633}{2^{13}}.$$

Suppose $\mathfrak{k} \in \{\rho(\mathfrak{m}), \rho^2(\mathfrak{m})\}$, then we reason as in the proof of Theorem 4.2 that

$$\kappa_-\theta_1 \geq \mu\theta_2 > \left(1 - \frac{1}{2^{p-1}}\right)\left(1 - \frac{f}{2^p}\right) > \left(1 - \frac{1}{2^8}\right)\left(1 - \frac{1}{16}\right) = \frac{7650}{2^{13}},$$

a contradiction.

Assume the conditions of part 5. Then

$$\mu\theta_2 < \frac{5 \cdot 2^{p-2} + 1}{2^p}\left(1 + \frac{f}{2^p}\right) < \left(\frac{5}{4} + \frac{1}{2^9}\right)\left(1 + \frac{1}{16}\right) = \frac{10897}{2^{13}}$$

and

$$\kappa_- \theta_1 > \frac{7 \cdot 2^{p-3} - 1}{2^p} \left(1 - \frac{f}{2^p}\right) > \left(\frac{7}{8} - \frac{1}{2^9}\right)\left(1 - \frac{1}{16}\right) = \frac{6705}{2^{13}}.$$

If $\mathfrak{k} = \rho^{-1}(\mathfrak{m})$, then $\mathfrak{m} = \rho^2(\mathfrak{k}_-)$, so $\mu\theta_2 > 2\kappa_-\theta_1$ by (2.3), contradicting the above inequalities. □

Lemma 4.5 *Let $D \in \mathbb{N}$ be squarefree with $D \equiv 3 \pmod 4$ and let $\mathfrak{k}$ be a reduced ideal, $\mathfrak{k} = (Q, P)$ and $\rho(\mathfrak{k}) = (Q_+, P_+)$. Then $Q \not\equiv Q_+ \pmod 4$.*

Proof: If $Q \equiv Q_+ \pmod 4$, then $QQ_+ \equiv 0$ or $1 \pmod 4$. By (2.1), $QQ_+ = D - P_+^2$; this is $3 \pmod 4$ if P_+ is even and $2 \pmod 4$ if P_+ is odd. □

Throughout our protocol, $\mathfrak{k}, \mathfrak{m}, \kappa, \kappa_-, \mu, \mu_-, k, k_-, m, m_-$ are as in Corollary 4.3 and its proof.

Protocol:

Alice and Bob publicly agree on

- a large squarefree positive integer $D \equiv 3 \pmod 4$;
- a reduced principal ideal $\mathfrak{r}$ in $\mathbb{Z}(\sqrt{D})$,
- a bound $B \in \mathbb{N}$ on the exponents.

Both precompute $p = \lceil \log_2(46B^2 \max\{16, \log_2 B\}) \rceil$.

Alice

- secretly generates $a \in \mathbb{N}$, $a \le B$;
- computes $(\mathfrak{a}, d_a) = \mathrm{EXP}(\mathfrak{r}, 2^p + 1, a)$, here $\mathfrak{a} = (Q_a, P_a)$;
- sends (Q_a, P_a, d_a) to Bob.

Bob

- secretly generates $b \in \mathbb{N}$, $b \le B$;
- computes $(\mathfrak{b}, d_b) = \mathrm{EXP}(\mathfrak{r}, 2^p + 1, b)$, here $\mathfrak{b} = (Q_b, P_b)$;
- sends (Q_b, P_b, d_b) to Alice.

Alice

- computes $(\mathfrak{k}, k) = \mathrm{EXP}(\mathfrak{b}, d_b, a)$, here $\mathfrak{k} = (Q, P)$; she also knows $(\rho^{-1}(\mathfrak{k}), k_-)$;
- sets $q \equiv Q \pmod 4$, $0 \le q \le 3$;
- sets
$$b_1 = \begin{cases} 0 & \text{if } k_- \le 7 \cdot 2^{p-3}, \\ 1 & \text{if } k_- > 7 \cdot 2^{p-3}, \end{cases} \quad b_2 = \begin{cases} 0 & \text{if } k \le 5 \cdot 2^{p-2}, \\ 1 & \text{if } k > 5 \cdot 2^{p-2}, \end{cases} \quad b_3 = \begin{cases} 0 & \text{if } k \le 7 \cdot 2^{p-2}, \\ 1 & \text{if } k > 7 \cdot 2^{p-2}, \end{cases}$$
- sends (b_1, b_2, b_3, q) to Bob.

Bob

- computes $(\mathfrak{m}, m) = \mathrm{EXP}(\mathfrak{a}, d_a, b)$.
- determines an ideal $\mathfrak{l} = (\tilde{Q}, \tilde{P})$ according to the following rules:

  if $b_2 = 0$ then
    if $b_1 = 0$ then
      if $m > 7 \cdot 2^{p-2}$ then set $\mathfrak{l} = \rho^{-2}(\mathfrak{m})$, else set $\mathfrak{l} = \rho^{-1}(\mathfrak{m})$
    else
      if $m > 5 \cdot 2^{p-2}$ then set $\mathfrak{l} = \rho^{-1}(\mathfrak{m})$, else set $\mathfrak{l} = \mathfrak{m}$
  else (in this case $b_2 = 1$)
    if $m > 5 \cdot 2^{p-2}$ then
      set $\mathfrak{l} = \mathfrak{m}$
    else
      if $b_3 = 0$ then set $\mathfrak{l} = \mathfrak{m}$, else set $\mathfrak{l} = \rho(\mathfrak{m})$.
- Sets $\mathfrak{k} = \mathfrak{l}$ if $\tilde{Q} \equiv Q \pmod 4$ and $\mathfrak{k} = \rho(\mathfrak{l})$ if $\tilde{Q} \not\equiv Q \pmod 4$.

We see that after both parties have completed their respective double exponentiations, Alice simply communicates five more bits to Bob. Upon receipts of these bits, Bob is able to determine the ideal $\mathfrak{k} = (Q, P)$ computed by Alice. The common key is a substring of P or Q of suitable size, where the last two bits of Q should not be used in the key. In the precomputation, $\mathfrak{r}$ can be obtained by applying a few iterations of (2.1) to the ideal $\mathcal{O} = (1, 0)$. Similarly, Bob can easily compute $\mathfrak{l}$ and $\mathfrak{k}$ using (2.1) and (2.4).

It remains to be shown that the ideal computed by Bob is in fact the same ideal $\mathfrak{k}$ that Alice generates.

Theorem 4.6 *After the protocol is executed, both communicants have computed the same ideal* $\mathfrak{k}$.

Proof: By Corollary 4.3, $\mathfrak{m} \in \{\rho^{-2}(\mathfrak{k}), \rho^{-1}(\mathfrak{k}), \mathfrak{k}, \rho(\mathfrak{k}), \rho^2(\mathfrak{k})\}$. If we show that if $\mathfrak{l} \in \{\mathfrak{k}, \rho^{-1}(\mathfrak{k})\}$, then by Lemma 4.5, $\mathfrak{k} = \mathfrak{l}$ if $\tilde{Q} \equiv Q \pmod 4$ and $\mathfrak{k} = \rho(\mathfrak{l})$ otherwise. We note that by Corollary 3.5, $k - 2 < d_1\kappa < k + 1$, $k_- - 2 < d_1\kappa_- < k_- + 1$, and $m - 2 < d_2\mu < m + 1$.

Suppose first that $b_2 = 0$, so $k \leq 5 \cdot 2^{p-2}$ and hence $d_1\kappa < 5 \cdot 2^{p-2} + 1$.

Case $b_1 = 0$. Then $d_1\kappa_- < 7 \cdot 2^{p-3} + 1$, so by part 1 of Lemma 4.4, $\mathfrak{k} \in \{\rho^{-2}(\mathfrak{m}), \rho^{-1}(\mathfrak{m}), \mathfrak{m}\}$. If $m > 7 \cdot 2^{p-2}$, then $d_2\mu > 7 \cdot 2^{p-2} - 1$, so by part 6 of the same lemma, $\mathfrak{k} \neq \mathfrak{m}$. Hence, $\mathfrak{k} \in \{\rho^{-2}(\mathfrak{m}), \rho^{-1}(\mathfrak{m})\}$ and $\mathfrak{l} = \rho^{-2}(\mathfrak{m}) \in \{\mathfrak{k}, \rho^{-1}(\mathfrak{k})\}$. Similarly, if $m \leq 7 \cdot 2^{p-2}$, then $d_2\mu < 7 \cdot 2^{p-2} + 1$, so again by part 6 of the same lemma $\mathfrak{k} \neq \rho^{-2}(\mathfrak{m})$. Hence, $\mathfrak{k} \in \{\rho^{-1}(\mathfrak{m}), \mathfrak{m}\}$ and $\mathfrak{l} = \rho^{-1}(\mathfrak{m}) \in \{\mathfrak{k}, \rho^{-1}(\mathfrak{k})\}$.

Case $b_1 = 1$. Then $d_1\kappa < 5 \cdot 2^{p-2} + 1$ and $d_1\kappa_- > 7 \cdot 2^{p-3} - 1$, so by parts 3 and 4 of Lemma 4.4, $\mathfrak{k} \in \{\rho^{-1}(\mathfrak{m}), \mathfrak{m}, \rho(\mathfrak{m})\}$. If $m > 5 \cdot 2^{p-2}$, then by part 2 of the lemma, $\mathfrak{k} \in \{\rho^{-1}(\mathfrak{m}), \mathfrak{m}\}$, and if $m \leq 5 \cdot 2^{p-2}$, then by part 5, $\mathfrak{k} \in \{\mathfrak{m}, \rho(\mathfrak{m})\}$. In either case, $\mathfrak{l} \in \{\mathfrak{k}, \rho^{-1}(\mathfrak{k})\}$.

Now suppose that $b_2 = 1$. Then part 2 of Lemma 4.4 yields $\mathfrak{k} \in \{\rho^2(\mathfrak{m}), \rho(\mathfrak{m}), \mathfrak{m}\}$. If $m > 5 \cdot 2^{p-2}$, then again by part 2, $\mathfrak{k} = \mathfrak{m} = \mathfrak{l}$. If $m \leq 5 \cdot 2^{p-2}$, then part 6 shows that $\mathfrak{k} \in \{\rho(\mathfrak{m}), \mathfrak{m}\}$ if $b_3 = 0$ and $\mathfrak{k} \in \{\rho^2(\mathfrak{m}), \rho(\mathfrak{m})\}$ if $b_3 = 1$. In either case, $\mathfrak{l} \in \{\mathfrak{k}, \rho^{-1}(\mathfrak{k})\}$. □

5 Implementation

The algorithms described above were implemented using the C++ computer algebra library LiDIA [10]. All computations in this section were performed using the GNU g++ compiler version 2.91.66 on a Pentium III 800 Mhz computer running Linux.

A number of optimizations have been incorporated into our implementation. In practice, algorithm MULT can be improved by testing for two commonly-occurring cases: squaring an ideal ($\gcd(Q_1/\sigma, Q_2/\sigma) = Q_1/\sigma$), and the case $\gcd(Q_1/\sigma, Q_2/\sigma) = 1$. Descriptions of the corresponding algorithms can be found in [2] and [5].

We now need to discuss how the value of s was selected. In Theorem 3.9 we need $d_0 < 9S^2\tilde{Q}2^{p-2}$, where $S^2\tilde{Q} \leq Q_1Q_2 < (2\sqrt{D})(2\sqrt{D}) = 4D$. Thus, if $d_0 < 9D2^p$, we have $d_0 < 9S^2\tilde{Q}2^{p-2}$. We also require $d_1 (= d_0)$ to satisfy $d_1/2^s < 1/10$ in Lemma 3.4. Thus, if $2^s > 90D2^p$, the EXP algorithm must work. However, in practice this is a much larger value of s than is actually needed.

Suppose we select some C and K such that

$$2^s > CK2^p/\sqrt{D}$$

and d_1 in Lemma 3.4 satisfies $d_1 < K2^p$. Although we have been at pains to point out how backward steps can be performed in our algorithms, it turns out that for large values of D we require very few backward steps. The reason for this is that the first reduced ideal $\mathfrak{b}_l$ that we find is such that $\Psi_l \leq 1$ (see Lemma 3.3). Indeed, it is often much smaller than 1. Since, as we shall see below, K can be taken to be small, we almost always have $d_1\Psi_l/S < 2^p$ and, as a consequence, we expect $d_1\hat{\Psi}_l/S < 2^p$ or $T_{l-2} \leq M$. If we now refer to the proof of Corollary 3.5, we may assume that

$$\frac{d_1\Psi_j}{S} - \frac{d_1\hat{\Psi}_j}{S} = \frac{\eta d_1 B_{j-2}}{Q_0 2^s S},$$

where $0 < \eta < 1$ and $j \leq k$. Since $B_{j-2} \leq B_{k-2}$, we get

$$\frac{B_{j-2}}{Q_0 2^s} \leq \frac{\Psi_k + \left|\overline{\Psi_k}\right|}{2^{s+1}\sqrt{D}} \leq \frac{\psi_{k-1}\Psi_{k-1} + 1}{2^{s+1}\sqrt{D}} < \frac{3\psi_{k-1}S/2 + 1}{2^{s+1}\sqrt{D}} < \frac{(5/4)\psi_{k-1}S}{2^s\sqrt{D}} \ .$$

It follows that

$$\frac{\eta d_1 B_{j-2}}{Q_0 2^s S} < \frac{(5/4)d_1}{2^s\sqrt{D}}\psi_{k-1} \ ;$$

thus, if $\psi_{k-1} < 4C/5$, then $d_1\Psi_j/S - d_1\hat{\Psi}_j/S < 1$. Also, since $B_{j-3} < B_{j-2}$, we get

$$\frac{d_1\hat{\Psi}_j}{S} < \frac{d_1\psi_{j-1}\Psi_{j-1}}{S} < \psi_{j-1}\left(\frac{d_1\hat{\Psi}_j}{S} + 1\right) \ ;$$

thus,

$$d_j < (2^p + 1)\psi_{j-1} + 1 < \psi_{j-1}(2^p + 2) \ .$$

Therefore, after executing step 1 of Algorithm MR we would have

$$e < \gamma_1\gamma_2\frac{(2^p + 2)^2}{2^p} + 1 < \frac{5}{4}\gamma_1\gamma_2 2^p \ .$$

Here $\gamma_1(>1)$ and $\gamma_2(>1)$ are complete quotients in the continued fraction expansion of ω. If $K > (5/4)\gamma_1\gamma_2$, then $e < K2^p$.

By the Gauß-Kuzmin law, we expect that if γ is any complete quotient in the continued fraction of any real number, then the approximate probability that $\gamma > b$ is $\log_2(1+1/b)$. Putting $C = \sqrt{5K/4}$, we see that the probability that $\gamma_1 > 4C/5$, $\gamma_2 > 4C/5$, or $\psi_{k-1} > 4C/5$ is very likely no more than $3\log_2(1+5/(4C)) < 15/(4C\log 2)$. If we put $C = (5/4)^{1/3}D^{1/12}$, we see that if D is large (>512 bits, say) we get $C > 2^{43}$, which means that it is very likely that we have $\psi_{k-1} < 4C/5$ and $e < K2^p$. Thus, if

$$2^s > \frac{2^p}{D^{1/4}},$$

$\hat{\Psi}_i$ should be a sufficiently good approximation of Ψ_i for the algorithms to work. Indeed, we found that values of s such that $2^s > D^{3/4}$ for $B = \sqrt{D}$ and $2^s > D^{1/4}$ for $B = D^{1/4}$ are perfectly adequate as long as D is large.

The reduction algorithm REDUCE primarily consists of expanding the continued fraction of $(P + \sqrt{D})/Q$, and hence we have used the more efficient Tenner's algorithm [21] for both the "forward" and "backward" reduction stages. In addition, we have tried two separate strategies to further speed the algorithm. In each iteration, $q_{j-1} = 1$ can be recognized easily by testing whether

$$Q_{j-1} \le P_{j-1} + \sqrt{D} < 2Q_{j-1} \ .$$

By handling this case specially, we avoid all multiplications and divisions for the computation of P_j and Q_j for that iteration. Since the Gauß-Kuzmin law predicts that $q_{j-1} = 1$ about 41% of the time, this should, and does, yield a fairly significant speed-up.

Our second method to speed REDUCE was to replace the simple continued fraction algorithm by the nearest integer continued fraction. The advantage of this algorithm is that the number of iterations required to find a reduced ideal is only about 70% of that using the simple continued fraction. However, the smaller number of iterations comes at the price of having $q_{j-1} \neq 1$ for all j, so we cannot combine our two methods. We found that using the nearest integer continued fraction for the forward steps and using the simple continued fraction while testing for partial quotients of 1 was slightly faster than other combinations of the two approaches, so we used it in our implementation. In the forward reduction steps, we compute T_{j-1} using the formula $T_{j-1} = q_{j-1}T_{j-2} - T_{j-3}$.

In order to test the efficiency of our protocol, we have computed numerous examples using discriminants of 512, 768, and 1024 bits. For each size of radicand, we have executed the key exchange protocol 100 times for each of 20 randomly chosen prime radicands of the given size. We have carried these computations out using exponent bounds $B = \sqrt{D}$ (with $2^s < D^{3/4}$) and $B = D^{1/4}$ (with $2^s < D^{1/4}$). In all cases, we take $\mathfrak{r} = \rho^5((1))$. Table 5.1 contains the average CPU time per communication partner for a single execution of the protocol, for each combination of the parameter sizes.

After profiling our implementation, we found that by far the most time-consuming part of the algorithm was ideal reduction — 97% compared to 3% for ideal multiplication. Clearly, any attempts to further optimize the protocol should begin there. One possible improvement would be to incorporate the NUCOMP algorithm of Shanks [19], which essentially combines the multiplication

Table 5.1: Average CPU time (in seconds) per key exchange per partner

$\log_2 D$	$B = D^{1/4}$, $2^s < D^{1/4}$	$B = \sqrt{D}$, $2^s < D^{3/4}$
512	0.48	1.20
768	1.33	2.90
1024	2.74	6.12

and reduction stages. In addition to keeping the sizes of the intermediate operands close to $\sqrt{D}$ (as opposed to operands close to D in Algorithm MULT), many of the reduction steps are replaced by slightly less expensive simple continued fraction steps.

One disadvantage of using the nearest integer continued fraction for reduction is that the backward steps in REDUCE are sometimes necessary, since some reduced ideals may be skipped in this type of continued fraction expansion. However, on average only 1 backward step was required per call to reduce, and the overall time was not greatly affected.

It should be noted that in all cases both partners do in fact end up with the same ideal, even though the precision used is less than what is theoretically necessary in the worst case. Thus, in practice, the second round of the protocol is never executed.

6 Security

6.1 Principal Ideal Testing

As shown in [17], our scheme can be broken by solving the principal ideal problem: given a reduced principal ideal $\mathfrak{a}$, find a generator μ, or $\log |\mu|$, such that $(\mu) = \mathfrak{a}$. It is also mentioned in [17] that solving the principal ideal problem is at least as difficult as factoring.

The best algorithms available for solving the principal ideal problem employ index-calculus techniques and are of subexponential complexity. Abel [1] was able show that under the Generalized Riemann Hypothesis (GRH) this problem has complexity

$$\exp\left((1.44 + o(1))\sqrt{\log D \log\log D}\right) \quad .$$

Improvements by Vollmer [20] suggest that the constant 1.44 can likely be reduced to $\sqrt{2} \approx 1.41$. To date, no algorithm of complexity $\exp\left((\log D)^{1/3}(\log\log D)^{2/3}\right)$ is known to exist.

The most efficient algorithm in practice is due to Jacobson [6], and is based on principles used in the self-initializing quadratic sieve factoring algorithm. Using this algorithm, it is possible to solve the principal ideal problem for values of $D \approx 10^{66}$ in a little over a day on a 298 MhZ UltraSPARC-II [6].

Notice that for all of these algorithms it is easy to verify unconditionally the correctness of a solution. The assumption of the GRH is needed to prove the complexity and the termination of the algorithms.

Currently, it appears that computing $R = \log \epsilon$, the regulator of K, is somewhat easier than solving the principal ideal problem in practice [5]. Computing the regulator is nevertheless a special case of solving the principal ideal problem ($\mathfrak{a} = (1)$), and hence security estimates based on the state of the art of regulator computation will yield conservative parameter selection guidelines.

Using a recent distributed implementation of the self-initializing quadratic sieve based algorithm in [5], we have successfully computed the regulators of a 90 decimal digit discriminant and a 101 decimal digit discriminant, both of which are significantly larger than those presented in [5]. Using a cluster of 16 550 Mhz Pentium III computers, we found that for the 90-digit

$$\begin{aligned} D = & 21522469810372840041048377124060167166863420091501850604626 3 \\ & 9189777165915901265583086318 04 \end{aligned}$$

we get (under the GRH)

$$\begin{aligned} R &= 131411783793381336054345076740506011516668614 4.03321787 \\ h &= 1 \ , \end{aligned}$$

where h denotes the ideal class number of K. This computation took a total of 10 days (approximately 5.2 months of CPU time). For the 101-digit

$$\begin{aligned} D = & 13022194102190350410319085329793205127319464132884776163361 5 \\ & 7836657137909258356026308739718466909983 6 \end{aligned}$$

we get (under the GRH)

$$\begin{aligned} R &= 3178025462317475553929176491549486361727631634782 60.945231457 \\ h &= 1 \ , \end{aligned}$$

and the entire computation on the cluster took 87 days (approximately 3.8 years of CPU time). More details about these computations will appear in a forthcoming paper.

6.2 The Choice of D

As demonstrated in [5, Chapter 7], the performance of the self-initializing quadratic sieve is highly sensitive to the quadratic character of D. In fact, the two discriminants listed above were constructed so that they would be especially amenable to this algorithm by forcing $(D/p) = 1$ for as many small primes p as possible. Conversely, selecting D such that $(D/p) = -1$ for as many small primes as possible will cause the regulator computation and the principal ideal problem to be much harder than average using this algorithm.

Table 6.2 contains run-times, taken from [5] and [7], for a number of various sized discriminants. The wide range of run-times between 60 and 63 digits and between 72 and 73 digits is due to

the varying quadratic characters of the discriminants. For example, the 73-digit discriminant was constructed so that $(D/p) = 1$ for all primes $p \leq 233$, and less than 14 hours of CPU time were required to compute its regulator. On the other hand, the 72-digit discriminant was constructed so that $(D/p) = -1$ for all primes $p \leq 337$, and approximately 18 days of CPU time were required to compute its regulator.

Table 6.2: Regulator computation CPU times for various sized discriminants

no. of digits in Δ	CPU time
54	6.26 m
58	17.79 m
60 — 63	49.56 m — 2.17 d
67	5.40 h
72 — 73	13.57 h — 18 d
80	2.05 d

The radicand D must also be selected such that there are sufficiently many reduced principal ideals to foil exhaustive search or baby-step giant-step attacks. Selecting D such that $(D/p) = -1$ for many small primes does have a minimizing effect on the product hR, but a result of Littlewood [11] states that under the GRH we still have $hR \gg \sqrt{D}/\log\log\Delta$. Thus, if we also force D to be prime, h will be odd and the Cohen-Lenstra heuristics predict that $h = 1$ about 75% of the time, so we will have $R \approx \sqrt{D}$.

Fortunately, constructing such values of D can be done quite expeditiously using special purpose sieve devices for solving systems of simultaneous linear congruences [7]. We used the Manitoba Scalable Sieve Unit [12] to generate three special values of D in this manner. The first 526-bit

$$\begin{aligned} D_1 =& 21400905888438118758623964453344480776164900535917750565452910904563090086200913324692607413981496938229384779456844137791200636936944043856483330455658015624 7 \end{aligned}$$

was constructed so that $(D_1/p) = -1$ for all odd $p < 587$. The 778-bit

$$\begin{aligned} D_2 =& 79666953182898317035577358353437500258218090254490555237895014476449334647283304758575357865473189110208001503375240570548565335970431641652924533814456596364099710031852691484279775282676190574931050546356285622857178373082218174494 3 \end{aligned}$$

has $(D_2/p) = -1$ for all odd $p < 727$, and the 1040-bit

$$\begin{aligned} D_3 =& 628422767086444915443603163898104886122604007365191407856568 \\ & 878324855316531764204362860362397355373970374319629714174971 \\ & 709266284883549779890009018542729063845063506367632958411625 \\ & 937373383256437213469372447649538329791081590593048268726182 \\ & 797367391958035401412654563687285881593903948899030520286997 \\ & 0057899153687 \end{aligned}$$

has $(D_3/p) = -1$ for all odd $p < 937$. All three D values are prime.

We have compared the speed of our protocol using these specially constructed D values with that using randomly selected prime D of the same sizes. For each of 20 random D values of the specified size and the special D value of that size, the key exchange protocol was performed 100 times. The average CPU time per key exchange for each communication partner are given in Table 6.3. As in the previous section we have used our LiDIA implementation on a Pentium III 800 Mhz computer running Linux. In all cases, we have used $B = D^{1/4}$, $2^s < D^{1/4}$, and $\mathfrak{r} = \rho^5((1))$. Even though the

Table 6.3: Protocol execution times (CPU seconds) for special D values

$\log_2 D$	Random	Special
526	0.49	0.49
778	1.29	1.31
1040	3.55	2.97

key exchange protocol does not execute any slower for these special radicands, using the state-of-the-art algorithm to solve the principal ideal problem will be significantly harder than for random radicands.

Acknowledgement

The authors greatfully acknowledge Detlef Hühnlein and Sachar Paulus for making an early version of their paper [4] available to us. We derived our idea of (f, p) representations from their manuscript.

References

[1] C.S. Abel, *Ein Algorithmus zur Berechnung der Klassenzahl und des Regulators reellquadratischer Ordnungen.* Ph.D. Thesis, Universität des Saarlandes, Saarbrücken, Germany, 1994.

[2] J. Buchmann, S. Düllmann, and H. C. Williams, On the complexity and efficiency of a new key exchange system. In *Advances in Cryptology - EUROCRYPT '89*, LNCS **434**, Springer (New York) 1990, 597–616.

[3] J. A. Buchmann and H. C. Williams, A key-exchange system based on imaginary quadratic fields. *J. Cryptology* **1** (1988), 107–118.

[4] D. Hühnlein and S. Paulus, On the implementation of cryptosystems based on real quadratic number fields. To appear in *Seventh Annual Workshop on Selected Areas in Cryptography - SAC2000*, Lecture Notes in Computer Science, Springer (New York), 2000.

[5] M. J. Jacobson, Jr., *Subexponential Class Group Computation in Quadratic Orders*. Ph.D. Thesis, Technische Universität Darmstadt, Darmstadt, Germany, 1999.

[6] M. J. Jacobson, Jr., Computing discrete logarithms in quadratic orders. *J. Cryptology*, **13** (2000), 473–492.

[7] M. J. Jacobson, Jr. and H. C. Williams, *New quadratic polynomials with high densities of prime values*. Submitted to *Math. Comp*.

[8] N. Koblitz, Elliptic curve cryptosystems. *Math. Comp.* **48** (1987), 203–209.

[9] N. Koblitz, Hyperelliptic cryptosystems. *J. Cryptology* **1** (1988), 94–97.

[10] The LiDIA Group, LiDIA: a C++ library for computational number theory. Software, Technische Universität Darmstadt, Germany, 1997. See http://www.informatik.tu-darmstadt.de/TI/LiDIA.

[11] J. E. Littlewood, On the class number of the corpus $P(\sqrt{-k})$. *Proc. London Math. Soc.* **27** (1928), 358–372.

[12] R. F. Lukes, C. D. Patterson, and H. C. Williams, Numerical sieving devices: Their history and some applications. *Nieuw Archief voor Wiskunde*, series 4, **13** (1995), 113–139.

[13] K. S. McCurley, A key distribution scheme based on factoring. *J. Cryptology* **1** (1988), 95–105.

[14] V. Miller, Use of elliptic curves in cryptography. In *Advances in Cryptology – Proceedings of CRYPTO '85*, LNCS **218**, Springer (New York) 1986, 417–426.

[15] R. W. K. Odoni, V. Varadharajan and P. W. Sanders, Public-key distribution in matrix rings. *Electr. Letters* **20** (1984), 386–387.

[16] H. J. J. te Riele, A. J. van der Poorten and H. C. Williams, Computer verification of the Ankeny-Artin-Chowla conjecture for all primes less than 100,000,000,000. To appear in *Math. Comp.*

[17] R. Scheidler, J. A. Buchmann and H. C. Williams, A key-exchange protocol using real quadratic fields. *J. Cryptology* **7** (1994), 171–199.

[18] R. Scheidler, A. Stein and H. C. Williams, Key-exchange in real quadratic congruence function fields. *Designs, Codes and Cryptography* **7** (1996), 153–174.

[19] D. Shanks, On Gauss and composition I, II. In *Proc. NATO ASI on Number Theory and Applications*, Kluwer Academic Press 1989, 163–179.

[20] U. Vollmer, Asymptotically fast discrete logarithms in quadratic number fields. In *Algorithmic Number Theory - ANTS4*, LNCS **1838**, Springer (Berlin) 2000, 581–594.

[21] H. C. Williams and M. C. Wunderlich, On the parallel generation of the residues for the continued fraction factoring algorithm. *Math. Comp.* **48** (1987), 405–423.