AN EXTREMELY SMALL AND EFFICIENT IDENTIFICATION SCHEME

William D. Banks[1], Daniel Lieman[2] and Igor E. Shparlinski[3]

[1] Department of Mathematics, University of Missouri
Columbia, MO 65211, USA
bbanks@math.missouri.edu

[2] Department of Mathematics, University of Missouri
Columbia, MO 65211, USA
lieman@math.missouri.edu

[3] Department of Computing, Macquarie University
Sydney, NSW 2109, Australia
igor@mpce.mq.edu.au

Abstract. We present a new identification scheme which is based on Legendre symbols modulo a certain hidden prime and which is naturally suited for low power, low memory applications.

1 Overview

One of the most desirable cryptographic functions is a secure, small, zero-knowledge public-key identification scheme. The applications are many and obvious – even the single application of credit/smart-card security is enough to stimulate research. In this paper, we present a scheme that requires extremely little computing power to perform a verification and to which we refer as FLIP (Fast Legendre Identification Protocol). Our scheme is unbalanced by design: the party proving his/her identity needs almost no computing power, while the party to whom the identity is being proved needs only a very small amount.

Our scheme is based on the assumption that integer factorization is a "hard problem." In fact, we believe that the *only* feasible attack on our scheme is via the factorization of a certain modulus M, hence the scheme is secure provided that M is reasonably large. In contrast to (say) **RSA** based signatures, our scheme offers the advantage that time consuming computation modulo M is required only for the *verifier* (it is not unreasonable to assume the *verifier* to be more powerful than the *prover*). This lends itself well to the credit/smart-card & bank paradigm.

We have conducted some preliminary tests using the interpreted number theory system PARI-GP. We find that FLIP (at high security) performs an identification as fast or faster than any other identification scheme of which we are aware, although key creation in FLIP is slower than in some other schemes, see [8].

We remark that although our scheme uses constructions similar to the Feige–Fiat–Shamir scheme, see [10], both schemes seem to be independent and rely on the hardness of different number theoretic problems (although the assumption of the hardness of integer factorization is common to both of them).

2 FLIP Description

Our construction is based on properties of the *Legendre* and *Jacobi* symbols.

We recall that given a prime p the Legendre symbol of a with $\gcd(a, p) = 1$ is defined as

$$\begin{cases} 1, & \text{if the congruence } a \equiv x^2 \pmod{p} \text{ is solvable,} \\ -1, & \text{otherwise.} \end{cases}$$

The Jacobi symbol modulo an odd integer k is defined as multiplicative extension of the Legendre symbol. That is,

$$\left(\frac{a}{k}\right) = \prod_{\nu=1}^{s} \left(\frac{a}{p_\nu}\right)^{e_\nu}$$

where

$$k = p_1^{e_1} \dots p_s^{e_s}$$

is the prime factorization of k.

We also recall the following basic properties of the Jacobi symbol (see Section 5.8 of [3]) which hold for any odd integer k and arbitrary integers l and r:

- ○ $\left(\frac{2}{k}\right) = (-1)^{(k^2-1)/8}$;
- ○ $\left(\frac{lr}{k}\right) = \left(\frac{l}{k}\right)\left(\frac{r}{k}\right)$;
- ○ $\left(\frac{l}{k}\right) = \left(\frac{r}{k}\right)$ if $l \equiv r \pmod{k}$;
- ○ $\left(\frac{l}{k}\right) = \left(\frac{k}{l}\right)(-1)^{(l-1)(k-1)/4}$, if l is odd.

The above properties provide very fast algorithms for computing Jacobi symbols, and thus guarantee the efficiency of our scheme.

FLIP has two formal security parameters, integers n and k.

For purposes of this paper, we will assume that *Irina* is proving her identity to the verifier *Victor*.

To create the signature *Irina* uses the following algorithm:

FLIP **initial set-up and key construction**

Step 1
Irina chooses two n-bit prime numbers p and r, and computes the product $M = pr$.

Step 2
Irina selects at random k relatively prime $2n$-bit integers a_j, computes the Legendre symbols

$$\alpha_j = \left(\frac{a_j}{p}\right), \qquad j = 1, \ldots, k,$$

and checks that at least one $\alpha_j = -1$.

Step 3
Irina publishes as her public key the product M and the collection of k pairs (a_j, α_j), $j = 1, \ldots, k$.

Step 4
Irina discards the value of r and retains as her private key the prime p.

The verification protocol has another security parameter which is a non-negative integer l.

The FLIP **verification sequence**

Step 1

Victor chooses l random $2n$-bit integers $s_1, \ldots, s_l$ and l subsets of the set $\{a_1, \ldots, a_k\}$, and for each subset, he computes the product of s_i^2 and the selected integers modulo M. In other words, *Victor* chooses l sets of k random bits $e_{ij} = 0, 1$, and computes for $i = 1, \ldots, l$

$$C_i \equiv s_i^2 \prod_{j=1}^{k} a_j^{e_{ij}} \pmod{M}, \qquad 0 \le C_i \le M - 1.$$

Step 2

Victor transmits the l numbers C_i, $i = 1, \ldots, l$.

Step 3

Irina computes and transmits the l Legendre symbols

$$\vartheta_i = \left(\frac{C_i}{p}\right), \qquad i = 1, \ldots, l.$$

Step 4

Victor verifies each of the l Legendre symbols transmitted by *Irina* is correct. That is, he verifies that

$$\vartheta_i = \prod_{j=1}^{k} \alpha_j^{e_{ij}}, \qquad i = 1, \ldots, l.$$

We first note that in terms of the parameters n, k and l, we have the following properties:

- ○ the bit size of the private key length is n
- ○ the bit size of the public key length is $2n + k(n + 1)$;
- ○ the total number of bits transmitted is $(2n + 1)l$.

We remark that the relatively prime numbers $a_1, \ldots, a_k$ need not be chosen by *Irina*. They can be globally available or even produced by *Victor*. In either of these cases, they are not strictly speaking a part of the public key, and the bit length of the public key drops to $2n+k$. In any case, the generation of the a_i is not time consuming. In fact even if one decides to select k random $2n$-bit prime numbers this can done be efficiently by selecting random integers in the interval $[2^{n-1}, 2^n - 1]$ and testing them for primality. Classical density results about the distribution of prime numbers and primality testing algorithms (see [3, 5, 6]) guarantee the efficiency of this procedure. The condition that at least one of $a_1, \ldots, a_k$ is not a quadratic residue modulo p is very easy to satisfy as well, for example, by selecting a_1 with this property.

It is also useful to recall that each computation of a Legendre symbol involved in this scheme takes $O(n^2)$ bit operations; see Theorem 5.9.3 of [3] or Section 1.4 of [5]. Each multiplication modulo M takes $O(n^2)$ bit operations if one uses naive arithmetic and $O(n \log n)$ bit operations if one uses fast arithmetic, see Theorems 7.8 and 8.5 of [1] or Theorem 8.24 and Corollary 9.9 of [6].

3 Security Analysis

It is obvious that the probability of impersonating a valid private key (that is, the probability of a correct guess of l individual Legendre symbols) is 2^{-l}. This is an "on-line" attack, and for attacks of this type, it is common to request the 2^{40} level of security. Thus the choice $l = 40$ will be satisfactory.

The probability that the same product will be used twice during N rounds of verification, thus allowing an attacker to collect and re-use *Irina*'s replies, is

$$P_{l,k,N} = 1 - \left(1 - 2^{-k}\right)^{Nl}.$$

In particular $P_{l,k,N} \sim Nl2^{-k}$ if $Nl2^{-k} \sim 0$. This is an "off-line" attack, and for attacks of this type, it is common to request the 2^{80} level of security. Assuming that $l = 40$ and that $N = 10000$ identification rounds are to be made, one can easily verify that the choice $k = 99$ guarantees

$$P_{l,k,N} \le 2^{-80}.$$

An "off-line" brute force attack (that is, correctly guessing a valid private key) would succeed with probability 2^{-k}.

One can also apply the above scheme with $s_1 = \ldots = s_l = 1$. However in this case a more sophisticated attack can be used. An attacker can precompute the products

$$\prod_{j=1}^{\lfloor k/2 \rfloor} a_j^{f_j} \qquad \text{and} \qquad \prod_{i=\lfloor k/2 \rfloor+1}^{k} a_j^{g_j}$$

for all binary vectors $\left(f_1, \ldots, f_{\lfloor k/2 \rfloor}\right)$ and $\left(g_{\lfloor k/2 \rfloor+1}, \ldots, g_k\right)$ and then try to find a representation of the challenges C_i, $i = 1, \ldots, l$, by looking at the precomputed values. This "meet in the middle" attack requires of order $2^{k/2}$ operations and the same amount of memory, so it is not likely to be efficient. In any case, using random squares in the computation of the challenges rules out this attack completely.

Another possible attack is via the known values of Legendre symbols. In theory, if one knows the Legendre symbols modulo p of the first $O(\log^2 p)$ integers, then this is enough to identify p uniquely (see [7]), but no one has been able to produce an efficient algorithm to accomplish this identification; indeed, the security schemes defined in [2] are based on the intractability of this problem. Note that in our scheme *Irina* does not verify that the numbers C_i, $i = 1, \ldots, l$, for which she is supposed to compute Legendre symbols are valid products of the integers a_j, $j = 1, \ldots, k$, used to construct the public key (indeed, such a verification would be infeasible). Thus, the attacker can force *Irina* to compute the Legendre symbol modulo p of any integer C. However, we believe that identifying the prime number p from the values of Legendre symbols modulo p of integers is completely infeasible.

Of course the attacker is able to compute

$$\left(\frac{C}{r}\right) = \left(\frac{C}{M}\right)\left(\frac{C}{p}\right)$$

for any C as well, but the same arguments as above apply to this as well.

As one of the advantages of our scheme we note that it is an *honest verifier zero-knowledge* scheme. That is, an honest verifier, who uses only "legimate" challenges C_i, i.e., challenges of the form

$$C_i \equiv s_i^2 \prod_{j=1}^{k} a_j^{e_{ij}} \pmod{M}, \qquad 0 \le C_i \le M - 1,$$

$i = 1, \ldots, l$, does not obtain any new information from the prover.

Finally, one could successfully attack this scheme by factoring either M or finding the representation of C_i as a product of powers of a_j. That is, by finding representations

$$C_i \equiv \prod_{j=1}^{k} a_j^{x_{ij}} \pmod M, \qquad i = 1, \ldots, l,$$

with integer x_{ij}. However, it is easy to see that the latter problem is not easier than the discrete logarithm problem. Indeed, even if $a_1, \ldots, a_k$ belong to a cyclic group $\mathcal{G}$ modulo M and even if representations $a_j \equiv g^{d_j} \pmod M$, $j = 1, \ldots, l$, are known, where g is a generator of $\mathcal{G}$, then finding a representation

$$C \equiv \prod_{j=1}^{k} a_j^{x_j} \equiv g^{x_1 d_1 + \ldots x_k d_k} \pmod M$$

is no easier than the general discrete logarithm problem modulo M. However, it has been shown in [9] that the discrete logarithm problem modulo a composite $M = pr$ (or even the possibly easier problem of breaking the Diffie–Hellman scheme) is as hard as factoring M; see also [4]. In particular, the prime numbers p and r should be selected to avoid the all known "short-cuts" in the integer factorization of $M = pr$ and in solving the discrete logarithm problem modulo M. Some conditions of this kind have been described in [9].

4 Possible Modifications

Instead of using quadratic characters one can use characters of higher orders, for example bi-quadratic characters. In this case a smaller value of l can be selected for the verification procedure, thus reducing the number of bits exchanged. For example, using bi-quadratic characters, one can use a value of l that is twice as small in order to provide the same level of security. More generally, characters of order d reduce this value by approximately $d/2$ times. On the other hand, the computational cost of computing higher order characters grows rather quickly with d. Nevertheless, our preliminary computational experiments have confirmed that bi-quadratic characters can be incorporated rather efficiently in this scheme.

Another possible modification may help to hide the values ϑ_i, $i = 1, \ldots, l$. In order to do so, *Irina* and *Victor* select some large integer weights w_i, $i = 1, \ldots, l$, (cooperatively, say each of them provides half of the bits of each element). Then *Irina* sends the sum $W = \vartheta_1 w_1 + \ldots + \vartheta_l w_l$ which can be verified by *Victor*. However finding the values of $\vartheta_1, \ldots, \vartheta_l$ from the value of W is equivalent to the knapsack problem which is known to be **NP**-complete. In fact, there is no need to select the weights w_i, $i = 1, \ldots, l$, for each round. They can be some initially agreed upon functions of ϑ_i, $i = 1, \ldots, l$. Moreover, if l is too small to guarantee the security of the knapsack problem, then *Irina* and *Victor* may use more weights w_i, $i = 1, \ldots, L$, with some $L \geq l$ and compute the sum

$$W = B_1(\vartheta_1, \ldots, \vartheta_l) w_1 + \ldots + B_L(\vartheta_1, \ldots, \vartheta_l) w_L$$

where B_i, $i = 1, \ldots, L$, are some Boolean functions of l variables. Probably one can even use $B_i(\vartheta_1, \ldots, \vartheta_l) = \vartheta_i$ for $i = 1, \ldots, l$.

References

1. A. V. Aho, J. E. Hopcroft and J.D. Ullman, *The design and analysis of computer algorithms*, Addison-Wesley, Reading, MA, 1975.
2. M. Anshel and D. Goldfeld, ‘Zeta functions, one-way functions, and pseudorandom number generators’, *Duke Math. J.*, **88** (1997), 371–390.
3. E. Bach and J. Shallit, *Algorithmic number theory*, MIT Press, 1996.
4. E. Biham, D. Boneh and O. Reingold, ‘Breaking generalized Diffie-Hellman modulo a composite is not weaker than factoring’, *Inform. Proc. Letters*, **70** (1999), 83–87.
5. H. Cohen *A course in Computational Algebraic Number Theory*, Springer-Verlag, Berlin, 1997.
6. J. von zur Gathen and J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, Cambridge, 1999.
7. D. Goldfeld and J. Hoffstein, ‘On the number of Fourier coefficients that determine a modular form’, *Contemp. Math.*, **143** (1993), 385–393.
8. J. Hoffstein, D. Lieman and J. Silverman, ‘Polynomial rings and efficient public key authentication’, *Proc. the Intern. Workshop on Cryptographic Techniques and E-Commerce (CrypTEC ’99)*, M. Blum and C.H. Lee, eds., City University of Hong Kong Press (to appear).
9. K. S. McCurley, ‘A key distribution system equivalent to factoring, *J. Cryptology*, **1** (1988), 95–105.
10. A. J. Menezes, P. C. van Oorschot and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, FL, 1996.