Electronic Preprint for Journal of Information Processing Vol.20 No.4

Regular Paper

Reducing Communication Complexity of Random Number Bitwise-Sharing for Efficient Multi-party Computation

Naoto Kiribuchi[1,†1] Ryo Kato[1] Takashi Nishide[2] Tsukasa Endo[3]
Hiroshi Yoshiura[1,a)]

Received: October 5, 2011, Accepted: May 12, 2012

Abstract: It is becoming more and more important to make use of personal or classified information while keeping it confidential. A promising tool for meeting this challenge is secure multi-party computation (MPC). However, one of the biggest problems with MPC is that it requires a vast amount of communication. We analyzed existing MPC protocols and found that the random number bitwise-sharing protocol used by many of them is notably inefficient. By devising a representation of the truth values and using special form prime numbers, we propose efficient random number bitwise-sharing protocols, dubbed "Extended-Range I and II," which reduce the communication complexity to approximately 1/6th that of the best of the existing such protocol. We reduced the communication complexity to approximately 1/26th by reducing the abort probability, thereby making previously necessary backup computation unnecessary. Using our improved protocol, "Lightweight Extended-Range II," we reduced the communication complexities of equality testing, comparison, interval testing, and bit-decomposition, all of which use the random number bitwise-sharing protocol, by approximately 91, 79, 67, and 23% (for 32-bit data), respectively. We also reduce the communication complexity of private exponentiation by about 70% (for 32-bit data and five parties).

Keywords: multi-party computation, secret sharing, Random Number Bitwise-Sharing

1. Introduction

Although gathering personal information (e.g., a person's age, address, and buying history) and using it directly or via data mining enable the provision of higher quality services, leakage of such information has become a serious problem. Similarly, while utilization of sensor logs can be profitable, their leakage is problematic. Moreover, in cloud computing, the confidentiality of remotely located personal or classified information must be guaranteed. It has thus become crucial to balance data availability against information confidentiality.

A promising tool for meeting this challenge is secure multi-party computation (MPC). Here we focus on MPC based on Shamir's (k, n) threshold secret sharing [10] in which a "share," or snippet, of secret information is distributed to n parties, and the parties can reconstruct the secret by gathering k shares.

MPC based on Shamir's scheme enables multiple parties to obtain the function value $f(s_1, \ldots, s_t)$ without revealing secrets $s_1, \ldots, s_t$. Various existing protocols (e.g., addition, multiplication, equality testing, and comparison) [2], [4], [5], [6], [7], [8], [11] enable the construction of the function for using personal or classified information, so they can be used to balance data availability against information confidentiality.

However, one of the biggest problems with MPC is that it requires a vast amount of communication and thus a vast amount of processing time. The multiplication protocol, which obtains the product of secrets $a, b \in \mathbb{Z}_p$, requires $n(n-1)$ times of communication since n parties communicate with each other. The complexity of MPC for protocols other than for multiplication, such as for comparison and equality testing, is evaluated in terms of the number of times the multiplication protocol is used. Much research has gone into making these other protocols more efficient by reducing the number of multiplications and parallelizing them [4], [6], [7], [8], [11]. For example, the comparison of two 32-bit secrets among five parties requires 36,640 multiplications and 44 rounds (number of parallel multiplications), resulting in approximately 2.8 MB of information being communicated [4]. Nishide and Ohta improved this protocol so that only 8,933 multiplications are required and 698 KB of information is communicated [8]. However, the protocol still requires a vast amount of communication, and parallelization of multiplications does not reduce the amount of information communicated. Hence it is required to make the protocols more efficient.

In this work, we analyzed the existing protocols and found that the random number bitwise-sharing protocol used by many of them is notably inefficient. For the comparison above, it ac-

[1] The University of Electro-Communications, Chofu, Tokyo 182–8585, Japan
[2] Kyushu University, Nishi, Fukuoka 819–0395, Japan
[3] Toshiba Corporation, Kawasaki, Kanagawa 212–8582, Japan
[†1] Presently with NTT Secure Platform Laboratories
[a)] yoshiura@hc.uec.ac.jp
A part of this work will be published in The 12th International Workshop on Information Security Applications (WISA 2011)

counts for 7,296 of the 8,933 multiplications. Though Toft improved this protocol [11] so that only 5,424 multiplications are required, it is still the dominant protocol. On the basis of our analysis, we construct five protocols for more efficient random number bitwise-sharing and use them to improve the efficiency of higher level protocols (such as comparison and equality testing) that use the random number bitwise-sharing protocol. Our protocols are based on two novel ideas.

The first idea is to use a new representation of the truth values. The random number bitwise-sharing protocol first generates candidates for ℓ-bit random number r in binary form. This random number must be less than the prime p used in the underlying secret sharing scheme. While existing protocols represent the result of the $r < p$ comparison as either 1 or 0, we remove inefficient bit operations by replacing these values with 0 and a non-zero value and using a special form prime number, p. On the basis of this idea, we construct two protocols, dubbed "Extended-Range I and II."

The second idea is to reduce the probability that the protocol aborts. If random number candidate r is not less than p, the protocol aborts and the parties retry. The abort probability for existing protocols is approximately $1/2$ in the worst case. To reduce this probability to less than $1/2^{\kappa}$, where κ is a predefined parameter, existing protocols generate alternative candidates. We improved our Extended-Range protocols so that they have an inherent abort probability of less than $1/2^{\kappa}$, making the generation of alternative candidates unnecessary. These improved protocols are dubbed "Lightweight Extended-Range I, II, and III."

All five of the proposed protocols reduce the communication complexity and the round complexity, i.e., the number of parallel multiplications, for equality testing, comparison, interval testing, and bit-decomposition, which use random number bitwise-sharing. The Lightweight Extended-Range II protocol, for example, reduces the communication complexities by about 91, 79, 67, and 23%, respectively (for 32-bit data). It also reduces the communication complexity of private exponentiation by about 70% (for 32-bit data and five parties). Our protocols are fundamental to sharing random number $r \in \mathbb{Z}_p$ in binary form and can be applied to other higher level protocols. We describe our protocols as they are applied to the "honest-but-curious" model. Application of standard techniques will make them more robust.

The organization of this paper is as follows. In Section 2 we analyze existing MPC protocols. In Section 3 we introduce our Extended-Range protocols and discuss their correctness, complexity, and security. In Section 4 we introduce our improved Lightweight Extended-Range protocols and discuss their complexity, correctness, and security. In Section 5 we describe the application of these protocols to higher level protocols and discuss their complexities. We conclude in Section 6 with a summary of the key points.

2. Related Work

2.1 Notation

- p: an ℓ-bit prime number
- $\mathbb{Z}_p$: a set of integer x where $0 \le x < p$
- $[a]$: a set of shares of secret $a \in \mathbb{Z}_p$
- $[a] + [b]$: shares of addition $[a + b \pmod p]$ where secrets $a, b \in \mathbb{Z}_p$
- $[a] \times [b]$: shares of multiplication $[a \times b \pmod p]$ where secrets $a, b \in \mathbb{Z}_p$
- $[a = b]$: shares of result of equality testing $a = b$ where secrets $a, b \in \mathbb{Z}_p$
- $[a < b]$: shares of result of comparison $a < b$ where secrets $a, b \in \mathbb{Z}_p$
- $[a_i]_B$: i'th bit shares of $a \in \mathbb{Z}_p$
- $[a]_B$: set of all bit shares of $a \in \mathbb{Z}_p$; i.e., $[a]_B = \{[a_i]_B | 0 \le i < \ell\}$

2.2 Shamir's (k, n) Threshold Secret Sharing

Given a secret $s \in \mathbb{Z}_p$, Shamir's (k, n) threshold secret sharing scheme [10] generates a polynomial,

$$f(x) = s + r_1 x + r_2 x^2 + \cdots + r_{k-1} x^{k-1} \pmod p, \qquad (1)$$

where $r_i \in \mathbb{Z}_p$ is a random number $(1 \le i \le k-1)$. Each of n parties P_d is given a share, $f(d)$ $(1 \le d \le n)$. To reconstruct the secret, the parties must gather k shares.

2.3 Multi-party Computation based on Secret Sharing Scheme

Because communication complexity is more dominant than local computational complexity, complexity for MPC is evaluated in terms of the communication complexity. Basic protocols for MPC are addition and multiplication. Given secrets $a, b \in \mathbb{Z}_p$, the addition protocol obtains $[c] = [a + b \pmod p]$ without revealing a, b. To compute $[c]$, each party simply adds $[a]$ and $[b]$ on $\mathbb{Z}_p$ independently. The complexity of the addition protocol is negligible since communication is unnecessary.

Given secrets $a, b \in \mathbb{Z}_p$, the multiplication protocol obtains $[c] = [a \times b \pmod p]$ without revealing a, b. The details are reported elsewhere [2], [5]. The communication complexity of the multiplication protocol is evaluated on the basis of the number of times the parties communicate with each other. As mentioned above, one invocation of the multiplication protocol requires $n(n-1)$ communications since n parties communicate with each other. However, if secret $a \in \mathbb{Z}_p$ and public value $e \in \mathbb{Z}_p$ are given, the computation of $[c] = [a \times e]$ requires no communication. Moreover, the complexity of computing any second-order polynomial, such as $[c] = [a^2 + b^2]$, is one multiplication operation [9].

Most existing MPC protocols use addition and multiplication, so their communication complexity is evaluated in terms of the number of multiplications required. The round complexity is also important in these protocols. We call protocols that use addition and multiplication "higher level protocols."

2.4 Problem of Multi-party Computation

One of the biggest problems with MPC is that it requires a vast amount of communication. The multiplication protocol requires $n(n-1)$ communications, and higher level protocols require much more communications.

Proposals have been made for improving each protocol. Damgård et al. [4] proposed bit-decomposition, which requires $94\ell \log_2 \ell + 93\ell$ multiplications and 38 rounds as well as

Table 1 Complexities of higher level protocols and of Joint Random Number Bitwise-Sharing (JRNBS) protocol used in them.

Protocol	Overall		JRNBS	
	Comm.	Rounds	Comm.	Rounds
Equality testing [8]	81ℓ	8	76ℓ	7
Comparison [8]	$279\ell + 5$	15	228ℓ	7
Interval testing [8]	$110\ell + 1$	13	76ℓ	7
Bit-Decomposition [11]	$31\ell \log_2 \ell + 71\ell + 30\sqrt{\ell}$	23	$52\ell + 24\sqrt{\ell}$	7
Private exponentiation [7]	$157\ell + (20n+36)\sqrt{\ell} + 10n + 8$	24	$128\ell + 24\sqrt{\ell}$	7

several protocols based on his improved bit-decomposition. Nishide et al. [8] improved several protocols by eliminating bit-decomposition, but they also reduced the complexity of bit-decomposition to $47\ell \log_2 \ell + 93\ell$ multiplications and 25 rounds. Recently, Toft [11] focused on and improved bit-decomposition so that it requires $31\ell \log_2 \ell + 71\ell + 30\sqrt{\ell}$ multiplications and 23 rounds. However, the protocols still require a vast amount of communication. For example, comparing two 32-bit secrets using the Nishide's protocol [8] requires 8,933 multiplications.

The complexities of random number sharing account for a substantial portion of the complexity of the higher level protocols. The Joint Random Number Bitwise-Sharing protocol is particularly dominant. For example, it accounts for 76ℓ multiplications and 7 rounds out of total 81ℓ multiplications and 8 rounds for equality testing. Even with Toft's improvement [11], the complexity is still $54\ell + 24\sqrt{\ell}$ multiplications and 7 rounds, and it is still dominant, as shown in **Table 1**. Cramer et al. proposed a protocol for pseudo-random secret sharing [3], but this protocol does not generate bitwise shares. Though most of the higher level protocols require both a random bitwise number and a random non-bitwise number, Cramer's protocol only cannot effectively reduce their complexity since the random bitwise number protocol is much more inefficient than the non-bitwise one.

2.5 Protocols for Random Number Sharing

2.5.1 Joint Random Number Sharing

This protocol generates shares $[r]$ where $r \in \mathbb{Z}_p$ is a uniformly random number [1]. Though this protocol contains no multiplication, the complexity is evaluated as 1 multiplication and 1 round since communication is necessary.

2.5.2 Joint Random Non-zero Sharing

This protocol generates shares $[r^*]$ where r^* is a uniformly non-zero random value. The complexity of this protocol is 3 multiplications and 2 rounds. The procedure of this protocol is as follows. First, the parties generate two sets of shares of uniformly random numbers $r_1, r_2 \in \mathbb{Z}_p$. Next, obtain $[s] = [r_1] \times [r_2]$ and reveal s. If $s = 0$, the parties retry. If $s \neq 0$, the parties output $[r_1]$.

2.5.3 Joint Random Bit Sharing

This protocol generates shares of a uniformly random bit $[r]$ where $r \in \{0, 1\}$. The complexity is 2 multiplications and 2 rounds [4].

2.5.4 Joint Random Number Bitwise-Sharing

This protocol generates bitwise shares of a uniformly random number $[r]_B$ where $r \in \mathbb{Z}_p$. Though required for various higher level protocols, this protocol is quite inefficient. The complexity is 76ℓ multiplications and 7 rounds [8].

The general procedure of this protocol is as follows. First, generate bitwise shares $[r]_B$ by applying Joint Random Bit Sharing ℓ times. Next, using the Bitwise Less-Than protocol, obtain the result of $r < p$ without revealing r, and then output $[r]_B$ if $r < p$. Otherwise, the parties retry.

The Bitwise Less-Than protocol accounts for 68ℓ multiplications and 7 rounds out of the total 76ℓ multiplications and 7 rounds of Joint Random Number Bitwise-Sharing.

The procedure of the Bitwise Less-Than protocol is as follows [4]. Let p_i be the i'th bit of ℓ-bit prime p.

(1) For $0 \leq i < \ell$, compute $[c_i] = [r_i \oplus p_i] = [r_i] + p_i - 2p_i[r_i]$.
(2) For each i, compute $[d_i] = \vee_{j=i}^{\ell-1}[c_j]$ using the Prefix-Or protocol [4], [8].
(3) For each i, compute $[e_i] = [d_i - d_{i+1}]$ where $[e_{\ell-1}] = [d_{\ell-1}]$.
(4) Compute $[r < p] = \sum_{i=0}^{\ell-1}(p_i \times [e_i])$.

Toft improved Joint Random Number Bitwise-Sharing and reduced its complexity to $52\ell + 24\sqrt{\ell}$ multiplications. However, the round complexity is still 7. In this work, we reduce the complexity further by making the Bitwise Less-Than protocol more efficient.

3. Extended-Range Protocols

Our proposed Joint Random Number Bitwise-Sharing protocols, "Extended-Range I and II," generate bitwise shares $[r]_B$ where r is a uniformly random number $r \in \mathbb{Z}_p$.

3.1 Key Ideas of Extended-Range Protocols

As shown in **Fig. 1**, the main procedure of the Extended-Range protocols is as follows.

(1) Generate bitwise shares $[r]_B$ by applying Joint Random Bit Sharing ℓ times.
(2) Obtain the result of $[z] = [r < p]$ by applying the Bitwise Less-Than protocol.
(3) Reveal z. If $z = 0$, output $[r]_B$. If z is a non-zero value, the parties retry.

The existing Bitwise Less-Than protocol outputs [1] if $r < p$, otherwise [0]. A key idea of the Extended-Range protocols is to replace the values output by the Bitwise Less-Than protocol with [0] and a non-zero value. Another is to assume that prime p satisfies some requirements that can still be practical in the real applications.

We focus on the second step (i.e., the Bitwise Less-Than protocol) since the other protocols used in the Extended-Range protocols are the same as in the existing protocols. Given ℓ bitwise shares $[r]_B$ and a public prime p, our Bitwise Less-Than protocol outputs [0] if $r < p$ and shares of a non-zero value otherwise without revealing r.

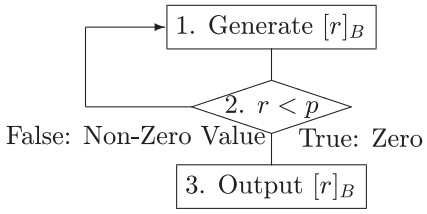

Fig. 1 Main procedure of Extended-Range protocols.

3.2 Extended-Range I

We use an ℓ-bit prime p that includes three 1s in its binary form. Since p is an ℓ-bit odd prime, the binary representation of p is as follows.

$$p : \underbrace{10\cdots010\cdots01}_{\ell \text{ bits}} \tag{2}$$

We use p_0, $p_{\ell-1}$, and p_m to denote LSB, MSB, and the remaining 1 bit between LSB and MSB. The output of the Bitwise Less-Than protocol is used in the Extended-Range I protocol as follows.

$$[r_{\ell-1}]_B \times \left\{ \left(\sum_{i=m+1}^{\ell-2} [r_i]_B \right) + [r_m]_B \times \left(\sum_{i=0}^{m-1} [r_i]_B \right) \right\} \tag{3}$$

The complexity of this formula is 2 multiplications and 2 rounds.

3.2.1 Correctness of Extended-Range I

The Bitwise Less-Than protocol used in the Extended-Range I protocol outputs [0] if $r < p$ and a share of a non-zero value if $r \not< p$. Let A and B designate ranges of bits in r.

$$p : 10\cdots010\cdots01 \tag{4}$$

$$r : *\underbrace{*\cdots**}_{A}\underbrace{*\cdots**}_{B} \tag{5}$$

If $r \not< p$, r is in at least one of the two following states in binary form.

$$p : 1\ 0\cdots010\cdots01 \tag{6}$$

$$1.\ r : 1\ \underbrace{\#\cdots\#}_{\textit{includes } 1}\ \underbrace{*\cdots\cdots*}_{\textit{arbitrary}} \tag{7}$$

$$2.\ r : 1\ \underbrace{*\cdots*}_{\textit{arbitrary}}\ 1\ \underbrace{\#\cdots\#}_{\textit{includes } 1} \tag{8}$$

We can translate these states into the following respective conditions.

(1) $r_{\ell-1} = 1$ and A includes 1.

(2) $r_{\ell-1} = r_m = 1$ and B includes 1.

For each condition, we can construct the following formulas that output a non-zero value if the condition is true and 0 if the condition is false.

(1) $r_{\ell-1} \times \sum_{i=m+1}^{\ell-2} r_i$

(2) $r_{\ell-1} \times r_m \times \sum_{i=0}^{m-1} r_i$

If $r < p$, these formulas output 0. If $r \not< p$, at least one formula outputs a non-zero value. Thus, by adding these formulas, we can obtain Eq. (9), which outputs 0 if $r < p$ and a non-zero value if $r \not< p$.

$$r_{\ell-1} \times \left\{ \left(\sum_{i=m+1}^{\ell-2} r_i \right) + r_m \times \left(\sum_{i=0}^{m-1} r_i \right) \right\} \tag{9}$$

The maximum output of Eq. (9) is $\ell-2$ and less than p. Hence we can translate Eq. (9) into Eq. (3), which is computed over $GF(p)$. We can see that Eq. (3) outputs [0] if $r < p$ and shares of a non-zero value if $r \not< p$ and that the output is correct.

3.3 Extended-Range II

The value of prime p that satisfies the assumption of the Extended-Range I protocol is mostly $p \equiv 1 \pmod 4$. However, there are protocols that require $p \equiv 3 \pmod 4$, such as some of Nishide's protocols [8]. Thus, we develop the Extended-Range II protocol for $p \equiv 3 \pmod 4$. Here we use a prime p that includes four 1 s in its binary form and $p_{\ell-1} = p_m = p_1 = p_0 = 1$, where $1 < m < \ell-1$. The binary representation of p is as follows.

$$p : \underbrace{10\cdots010\cdots011}_{\ell \text{ bits}} \tag{10}$$

The output of the Bitwise Less-Than protocol used in Extended-Range II is as follows.

$$[r_{\ell-1}]_B \left(\sum_{i=m+1}^{\ell-2} [r_i]_B \right) + ([r_{\ell-1}]_B [r_m]_B) \left\{ \left(\sum_{i=2}^{m-1} [r_i]_B \right) + [r_1]_B [r_0]_B \right\} \tag{11}$$

To obtain the output, the parties compute $[r_{\ell-1}]_B \times [r_m]_B$ and $[r_1]_B \times [r_0]_B$ in parallel. This makes the formula second-order, so its complexity is 1 multiplication and 1 round. Hence the total complexity of Eq. (11) is 3 multiplications and 2 rounds.

3.3.1 Correctness of Extended-Range II

The Bitwise Less-Than protocol used in Extended-Range II outputs [0] if $r < p$ and shares of a non-zero value if $r \not< p$. If $r \not< p$, r is in at least one of the three following states in binary form.

$$p : 1\ 0\cdots010\cdots011 \tag{12}$$

$$1.\ r : 1\ \underbrace{\#\cdots\#}_{\textit{includes } 1}\ \underbrace{*\cdots\cdots*}_{\textit{arbitrary}} \tag{13}$$

$$2.\ r : 1\ \underbrace{*\cdots*}_{\textit{arbitrary}}\ 1\ \underbrace{\#\cdots\#}_{\textit{includes } 1}** \tag{14}$$

$$3.\ r : 1\ \underbrace{*\cdots*}_{\textit{arbitrary}}\ 1\ \underbrace{*\cdots*}_{\textit{arbitrary}}\ 1\ 1 \tag{15}$$

For each state, we can construct a equation that outputs a non-zero value if r is in the state and 0 if r is not in the state.

(1) $r_{\ell-1} \times \sum_{i=m+1}^{\ell-2} r_i$

(2) $r_{\ell-1} \times r_m \times \sum_{i=2}^{m-1} r_i$

(3) $r_{\ell-1} \times r_m \times r_1 \times r_0$

If $r < p$, these formulas output 0. If $r \not< p$, at least one equation outputs a non-zero value. Thus, by adding these formulas, we can obtain Eq. (16), which outputs 0 if $r < p$, and a non-zero value if $r \not< p$.

$$r_{\ell-1} \times \left(\sum_{i=m+1}^{\ell-2} r_i \right) + (r_{\ell-1} \times r_m) \times \left\{ \left(\sum_{i=2}^{m-1} r_i \right) + r_1 \times r_0 \right\} \tag{16}$$

The maximum output of Eq. (16) is $\ell-3$ and less than p. Hence we can translate Eq. (16) into Eq. (11), which is computed over $GF(p)$. We can see that Eq. (11) outputs [0] if $r < p$ and shares of a non-zero value if $r \not< p$ and that the output is correct.

3.4 Complexity of Extended-Range Protocols

The Extended-Range protocols generate ℓ sets of bitwise shares $[r_i]_B$ in parallel through Joint Random Bit Sharing. The complexity of this step is 2ℓ multiplications and 2 rounds. Then the parties obtain the result of $r < p$ from the Bitwise Less-Than protocol. The complexity of this step is 2 multiplications and 2 rounds for Extended-Range I and 3 multiplications and 2 rounds for Extended-Range II. Thus, the complexity to generate one random number candidate and obtain the result of $r < p$ is $2\ell + 2$ multiplications and 4 rounds for Extended-Range I and $2\ell + 3$ multiplications and 4 rounds for Extended-Range II.

Since the abort probability for existing protocols is approximately $1/2$ in the worst case, these protocols generate four random number candidates [4], [8], [11]. Because the abort probability for the Extended-Range protocols is the same as that of the existing protocols, we evaluate the complexities for Extended-Range in the case of generating four candidates in parallel. The complexity is $8\ell + 8$ multiplications and 4 rounds for Extended-Range I and $8\ell + 12$ multiplications and 4 rounds for Extended-Range II (**Table 2**).

3.5 Security of Extended-Range Protocols

The existing Joint Random Number Bitwise-Sharing protocol [8], which outputs shares of random number r, satisfies the following conditions.

(1) The random number r can be any value x such that $0 \leq x < p$ with equal probability.
(2) Any information about r is not leaked unless k shares are collected.

If these two conditions are satisfied, the existing Joint Random Number Bitwise- Sharing protocol can be securely used in higher level protocols.

In the Extended-Range protocols, the steps to generate candidates for a random number are the same as those of the existing Joint Random Number Bitwise-Sharing protocol. Both the output of the existing protocol and those in the Extended-Range protocols are chosen from the candidates only by checking whether r is less than p. Thus, the Extended-Range protocols satisfy condition 1 above regardless of the choice of the prime number p.

The proposed Bitwise Less-Than protocols, which are used in the Extended- Range protocols, output 0 if $r < p$ and the output does not leak any information about r. If $r \not< p$, r is abandoned, and the value of the output causes no problem. Thus, the Extended-Range protocols do not leak any information about r and therefore satisfy condition 2 above. Since the Extended-Range protocols satisfy the conditions for the existing protocol, they can be securely used in higher level protocols. Note that any prime number can be used for the underlying secret sharing scheme, so the choice of p does not affect the security of the secret sharing scheme. Therefore, using a special form prime number does not compromise the security.

Table 2 Complexities of Random Number Bitwise-Sharing protocols.

Protocol	Comm.	Rounds
Nishide and Ohta [8]	76ℓ	7
Toft [11]	$52\ell + 24\sqrt{\ell}$	7
Extended-Range I	$8\ell + 8$	4
Extended-Range II	$8\ell + 12$	4
Lightweight Extended-Range I	$2\ell + 4$	3
Lightweight Extended-Range II	$2\ell + 5$	4
Lightweight Extended-Range III	$2\ell + 7$	5

4. Improvement of Extended-Range Protocols

Although the Extended-Range protocols are efficient compared to existing protocols [8], [11], the abort probability is still $1/2$ in the worst case. In this section, we introduce our more efficient protocols "Lightweight Extended-Range" protocols, which make the abort probability negligible.

The Lightweight Extended-Range I protocol uses a Mersenne prime. It is efficient but not practical since a Mersenne prime is sparse. The Lightweight Extended-Range II protocol uses a modified Mersenne prime, a "semi-Mersenne prime," which is sufficiently abundant. This protocol is thus practical as well as efficient. The Lightweight Extended-Range III protocol is a variation of these two protocols. It uses a prime number p that satisfies $p \equiv 1 \pmod 4$, whereas Lightweight Extended-Range I and II normally use a prime number $p \equiv 3 \pmod 4$.

4.1 Key Idea of Lightweight Extended-Range Protocols

The Lightweight Extended-Range protocols use a Mersenne or semi-Mersenne prime to make the abort probability negligible. However, the straightforward use of these primes would increase the protocol's complexity. The key idea of Lightweight Extended-Range is making the abort probability negligible without increasing the complexity by exchanging the truth values. These protocols use a non-zero value if $r < p$ and 0 if $r \not< p$ whereas the Extended-Range protocols use 0 if $r < p$ and a non-zero value if $r \not< p$. As shown in **Fig. 2**, the main procedure of the Lightweight Extended-Range protocols is as follows.

(1) Generate bitwise shares $[r]_B$ by applying Joint Random Bit Sharing protocol ℓ times and generate a share of a non-zero value $[r^*]$ by applying Joint Random Non-zero Sharing protocol in parallel.
(2) Obtain $[s] = [r < p]$ by applying Bitwise Less-Than protocol and reveal $[s \times r^*]$.
(3) If $s \times r^*$ is non-zero (i.e., $r < p$), output $[r]_B$. If $s \times r^* = 0$ (i.e., $r \not< p$), the parties retry.

Again we focus on the Bitwise Less-Than protocol used in the second step since the other protocols used in the Lightweight Extended-Range protocols are the same as those in the existing protocols. Given bitwise shares of an ℓ-bit random number $[r]_B$ and a public prime p, our Bitwise Less-Than protocol outputs shares of a non-zero value if $r < p$ and [0] otherwise without revealing r.

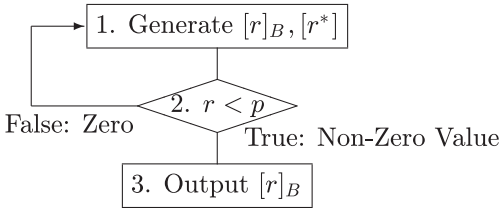

Fig. 2 Main procedure of Lightweight Extended-Range protocols.

4.2 Lightweight Extended-Range I

A Mersenne prime is a prime p such that $p = 2^\ell - 1$; i.e., the binary representation of an ℓ-bit Mersenne prime is as follows.

$$p = (\underbrace{1 \dots 1}_{\ell \text{ bits}})_2 \tag{17}$$

The output of the Bitwise Less-Than protocol used in Lightweight Extended-Range I is as follows.

$$[r^*]([r_0] + [r_1] + \cdots + [r_{\ell-2}] + [r_{\ell-1}] - \ell) \tag{18}$$

The complexity of Eq. (18) is 1 multiplication and 1 round.

4.2.1 Correctness of Lightweight Extended-Range I

If $r \neq p$, all r_i's are 1 since p is a Mersenne prime ($0 \leq i < \ell$). Thus, Eq. (18) obviously outputs [0]. If $r < p$, the output is shares of a non-zero value since $(r_0 + \cdots + r_{\ell-1} - \ell)$ is non-zero. Thus, we can see that the output is correct.

4.3 Lightweight Extended-Range II

Using a Mersenne prime is hardly practical since it is sparse, as noted above. Thus, we define a semi-Mersenne prime, which is sufficiently abundant.

A semi-Mersenne prime is an ℓ-bit prime such that

$$p = 2^\ell - 1 - 2^c, \tag{19}$$

where $0 < c < \ell - 1$.

The binary representation of a semi-Mersenne prime is

$$p = (1 \dots 101 \dots 1)_2 \tag{20}$$

and $p_c = 0$.

The output of the Bitwise Less-Than protocol used in Lightweight Extended-Range II is as follows.

$$[r^*]\left\{\sum_{i=c}^{\ell-1}[\overline{r_i}]\right\}\left\{\left(\sum_{i=c+1}^{\ell-1}[\overline{r_i}]\right) + [r_c] + \left(\sum_{i=0}^{c-1}[\overline{r_i}]\right)\right\} \tag{21}$$

Note that a bitwise NOT can be computed as $[\overline{r_i}] = 1 - [r_i]$. The complexity of Eq. (21) is 2 multiplications and 2 rounds.

4.3.1 Correctness of Lightweight Extended-Range II

If $r \neq p$, r is in either one of the two following states in binary form.

$$p : 1 \cdots 1\ 0\ 1 \cdots\cdots 1 \tag{22}$$

$$1.\ r : 1 \cdots\cdots 1\ \underbrace{* \cdots\cdots *}_{arbitrary} \tag{23}$$

$$2.\ r : 1 \cdots 1\ 0\ 1 \cdots\cdots 1 \tag{24}$$

We can translate these states into the following respective conditions.

(1) $r_{\ell-1}, \dots, r_c = 1$.

(2) $r_{\ell-1}, \dots, r_{c+1} = 1$ and $r_c = 0$ and $r_{c-1}, \dots, r_0 = 1$.

For each condition, we can construct a formula that outputs 0 if the condition is true and a non-zero value if the condition is false.

(1) $\sum_{i=c}^{\ell-1} \overline{r_i}$

(2) $(\sum_{i=c+1}^{\ell-1} \overline{r_i}) + r_c + (\sum_{i=0}^{c-1} \overline{r_i})$

Both of these formulas output a non-zero value if $r < p$. If $r \neq p$, one of them outputs 0. By multiplying these equations, we can obtain Eq. (21), which outputs shares of a non-zero value if $r < p$ and [0] if $r \neq p$.

4.4 Lightweight Extended-Range III

Almost all values of prime p that satisfy the condition for the Lightweight Extended-Range II protocol are in the form of $p \equiv 3 \pmod 4$. However, some protocols require $p \equiv 1 \pmod 4$, such as some of Nishide's protocols [8]. Thus, we construct the Lightweight Extended-Range III protocol for $p \equiv 1 \pmod 4$. Here we use a prime p that includes three 0 s in its binary form, and $p_{c+1} = p_c = p_1 = 0$, where $1 < c < \ell - 2$. The binary representation of p is as follows.

$$p = (1 \dots 1001 \dots 101)_2 \tag{25}$$

The output of the Bitwise Less-Than protocol used in Lightweight Extended-Range III is as follows.

$$[r^*]\left\{\left(\sum_{i=c+2}^{\ell-1}[\overline{r_i}]\right) + [\overline{r_{c+1}}][\overline{r_c}]\right\} \times \left\{\left(\sum_{i=c+2}^{\ell-1}[\overline{r_i}]\right) + [r_{c+1}] + [r_c] + \left(\sum_{i=2}^{c-1}[\overline{r_i}]\right) + [\overline{r_1}][\overline{r_0}]\right\} \tag{26}$$

The complexity of Eq. (26) is 4 multiplications and 3 rounds.

The Lightweight Extended-Range III protocol improves the generality of the Lightweight Extended-Range protocols, so we can use both $p \equiv 1 \pmod 4$ and $p \equiv 3 \pmod 4$.

4.4.1 Correctness of Lightweight Extended-Range III

If $r \neq p$, r is in either one of the two following states in binary form.

$$p : 1 \cdots 1\ \ 0\ 0\ \ 1 \cdots 1\ \ 0\ 1 \tag{27}$$

$$1.\ r : 1 \cdots 1\ \underbrace{\#\ \#}_{includes\ 1}\ \underbrace{* \cdots\cdots\cdots *}_{arbitrary} \tag{28}$$

$$2.\ r : 1 \cdots 1\ \ 0\ 0\ \ 1 \cdots 1\ \underbrace{\#\ \#}_{includes\ 1} \tag{29}$$

We can translate these states into the following respective conditions.

(1) $r_{\ell-1}, \dots, r_{c+2} = 1$, and ($r_{c+1} = 1$ or $r_c = 1$)

(2) $r_{\ell-1}, \dots, r_{c+2}, r_{c-1}, \dots, r_2 = 1$, and $r_{c+1} = r_c = 0$, and ($r_1 = 1$ or $r_0 = 1$)

For each condition, we can construct a formula that outputs 0 if the condition is true and a non-zero value if the condition is false.

(1) $(\sum_{i=c+2}^{\ell-1} \overline{r_i}) + \overline{r_{c+1}} \times \overline{r_c}$

(2) $(\sum_{i=c+2}^{\ell-1} \overline{r_i}) + r_{c+1} + r_c + (\sum_{i=2}^{c-1} \overline{r_i}) + \overline{r_1} \times \overline{r_0}$

Both of the equations output a non-zero value if $r < p$. If $r \neq p$, one of them outputs 0. By multiplying these equations, we can obtain Eq. (26), which outputs shares of a non-zero value if $r < p$ and [0] if $r \neq p$.

4.5 Complexity of Lightweight Extended-Range Protocols

The Lightweight Extended-Range protocols generate ℓ sets of bitwise shares $[r_i]_B$ and non-zero shares $[r^*]$ in parallel by Joint Random Bit Sharing and Joint Random Non-zero Sharing, respectively. The complexity of this step is $2\ell + 3$ multiplications and 2 rounds. The parties obtain the result of $r < p$ from the Bitwise Less-Than protocol. The complexity of this step is 1 multiplication and 1 round for Lightweight Extended-Range I, 2 multiplications and 2 rounds for Lightweight Extended-Range II, and 4 multiplications and 3 rounds for Lightweight Extended-Range

Ⅲ. Thus, the complexity to generate one random number candidate and obtain the result of $r < p$ is $2\ell + 4$ multiplications and 3 rounds for Lightweight Extended-Range I, $2\ell + 5$ multiplications and 4 rounds for Lightweight Extended-Range Ⅱ, and $2\ell + 7$ multiplications and 5 rounds for Lightweight Extended-Range Ⅲ.

As mentioned, the abort probability for the previous protocols is approximately $1/2$ in the worst case. To reduce this to $1/2^{\kappa}$, they generate four random number candidates where κ is a predefined parameter [4], [8], [11]*1. Meanwhile, the inherent abort probability of the Lightweight Extended-Range protocols is $1/2^{\kappa}$. In the case of Lightweight Extended-Range I, which uses a Mersenne prime $p = 2^{\ell} - 1$, $\kappa = \ell$. For example, Mersenne prime $p = 2^{31} - 1$ gives an abort probability of $1/2^{31}$. In the case of Lightweight Extended-Range Ⅱ, which uses a semi-Mersenne prime $p = 2^{\ell} - 1 - 2^{c}$, $\kappa > \ell - c - 1$. For example, semi-Mersenne prime $p = 2^{32} - 1 - 2^{2}$ gives an abort probability of less than $1/2^{29}$. In the case of Lightweight Extended-Range Ⅲ, which uses a prime $p = 2^{\ell} - 1 - 2^{c+1} - 2^{c} - 2^{1}$, the abort probability is $\kappa > \ell - c - 2$. For example, a prime $p = 2^{32} - 1 - 2^{6} - 2^{5} - 2^{1}$ gives an abort probability of less than $1/2^{25}$. Thus, the Lightweight Extended-Range protocols do not need to generate four candidates, which means that we can evaluate their complexities using only one candidate. The complexities of the Random Number Bitwise-Sharing protocols are summarized in Table 2.

4.6 Security of Lightweight Extended-Range Protocols

As mentioned, the existing Joint Random Number Bitwise-Sharing protocol [8], which outputs shares of random number r, satisfies the following conditions.

(1) The random number r can be any value x such that $0 \le x < p$ with equal probability.

(2) Any information about r is not leaked unless k shares are collected.

If these two conditions are satisfied, the existing Joint Random Number Bitwise- Sharing protocol can be securely used in higher level protocols.

In the same way as the Extended-Range protocols in Section 3.5, the Lightweight Extended-Range protocols satisfy condition 1 above regardless of the choice of prime number p.

The proposed Bitwise Less-Than protocols, which are used in the Lightweight Extended-Range protocols, output a non-zero value if $r < p$. Since this non-zero value is masked by a random number r^*, the output does not leak any information about r. If $r \not< p$, r is abandoned, and the value of the output causes no problem. Thus, the Lightweight Extended-Range protocols do not leak any information about r and therefore satisfy condition 2 above. Since the Lightweight Extended- Range protocols satisfy the conditions for the existing protocol, they can be securely used in higher level protocols. Note that using a special form prime number does not compromise the security as discussed in Section 3.5.

*1 The generation of four candidates is further discussed elsewhere [4].

Table 3 The number of prime numbers satisfying conditions for proposed protocols.

	Extended-Range		Lightweight Extended-Range		
ℓ	I	Ⅱ	I	Ⅱ	Ⅲ
16	4	3	0	3	3
32	6	2	0	7	6
64	2	5	0	3	0
128	6	5	0	4	8
256	4	2	0	2	5
512	5	2	0	6	4
1,024	3	5	0	9	2

4.7 Practicality of Proposed Protocols

Given bit-length ℓ, if there exists an ℓ-bit prime number p that satisfies the condition for a proposed protocol, the proposed protocols can be used to generate shares of a random number r more efficiently than the existing protocols. As shown in **Table 3**, there are prime numbers that satisfy the condition for the Extended-Range I and Ⅱ and the Lightweight Extended-Range Ⅱ protocol where $\ell = 16, 32, 64, 128, 256, 512$, and 1,024. For the Lightweight Extended-Range Ⅲ protocol as well, there are prime numbers that satisfy the condition except for $\ell = 64$. Note that we can use a 65-bit prime number that satisfies the condition instead of a 64-bit prime number. Only the Lightweight Extended-Range I protocol lacks the prime numbers that satisfy its condition, so it can only be used for the limited ℓ's (e.g., $\ell = 31$ and 61).

Therefore, the proposed protocols except for Lightweight Extended-Range I are sufficiently practical.

5. Application to Higher Level Protocols

We applied our five proposed protocols to equality testing [8], comparison [8], interval testing [8], bit-decomposition [11], and private exponentiation [7]. We briefly explain each of these higher level protocols. They are described in more detail elsewhere [7], [8], [11]. Given sets of shares $[a], [b]$, where $a, b \in \mathbb{Z}_p$, the equality testing protocol [8] and the comparison protocol [8] respectively obtain a set of shares $[a = b]$ and $[a < b]$ without revealing a and b. Given a set of shares $[a]$, where $a \in \mathbb{Z}_p$ and public values $c_1, c_2 \in \mathbb{Z}_p$, the interval testing protocol [8] obtains a set of shares $[c_1 < a < c_2]$ without revealing a. Given a set of shares $[a]$, where $a \in \mathbb{Z}_p$, the bit-decomposition protocol [11] obtains sets of bitwise shares $[a]_B$ without revealing any a_i. Given sets of shares $[a], [x]$, where $a, x \in \mathbb{Z}_p$, the private exponentiation protocol [7] obtains a set of shares $[x^a \bmod p]$ without revealing a and x.

The proposed protocols can be applied to these higher level protocols simply by replacing the existing Joint Random Number Bitwise-Sharing protocol with a proposed one. In the case of equality testing, the existing Joint Random Number Bitwise-Sharing protocol accounts for 76ℓ of 81ℓ multiplications and 7 of 8 rounds [8]. The proposed protocols reduce the communication and round complexities as shown in **Table 4**. The complexities of comparison, interval testing, bit-decomposition, and private exponentiation are respectively shown in **Tables 5**, **6**, **7**, and **8**. Note that the existing higher level protocols use various protocols for random number sharing.

Table 4 Complexities of equality testing protocols and of Joint Random Number Bitwise-Sharing protocols used in them.

Protocol		Random number sharing		Equality testing	
		Comm.	Rounds	Comm.	Rounds
Nishide and Ohta [8]		76ℓ	7	81ℓ	8
Extended-	I	$8\ell+8$	4	$13\ell+8$	5
Range	II	$8\ell+12$	4	$13\ell+12$	5
Lightweight	I	$2\ell+4$	3	$7\ell+4$	4
Extended-	II	$2\ell+5$	4	$7\ell+5$	5
Range	III	$2\ell+7$	5	$7\ell+7$	6

Table 5 Complexities of comparison protocols and of Joint Random Number Bitwise-Sharing protocols used in them.

Protocol		Random number sharing		Comparison	
		Comm.	Rounds	Comm.	Rounds
Nishide and Ohta [8]		228ℓ	7	$279\ell+5$	15
Extended-	I	$24\ell+24$	4	$75\ell+29$	12
Range	II	$24\ell+36$	4	$75\ell+41$	12
Lightweight	I	$6\ell+12$	3	$57\ell+17$	11
Extended-	II	$6\ell+15$	4	$57\ell+20$	12
Range	III	$6\ell+21$	5	$57\ell+26$	13

Table 6 Complexities of interval testing protocols and of Joint Random Number Bitwise-Sharing protocols used in them.

Protocol		Random number sharing		Interval testing	
		Comm.	Rounds	Comm.	Rounds
Nishide and Ohta [8]		76ℓ	7	$110\ell+1$	13
Extended-	I	$8\ell+8$	4	$42\ell+9$	10
Range	II	$8\ell+12$	4	$42\ell+13$	10
Lightweight	I	$2\ell+4$	3	$36\ell+5$	9
Extended-	II	$2\ell+5$	4	$36\ell+6$	10
Range	III	$2\ell+7$	5	$36\ell+8$	11

Table 7 Complexities of bit-decomposition protocols and of Joint Random Number Bitwise-Sharing protocols used in them.

Protocol		Random number sharing		Bit-Decomposition	
		Comm.	Rounds	Comm.	Rounds
Toft [11]		$52\ell+24\sqrt{\ell}$	7	$31\ell\log_2\ell+71\ell+30\sqrt{\ell}$	23
Extended-	I	$8\ell+8$	4	$31\ell\log_2\ell+27\ell+6\sqrt{\ell}+8$	20
Range	II	$8\ell+12$	4	$31\ell\log_2\ell+27\ell+6\sqrt{\ell}+12$	20
Lightweight	I	$2\ell+4$	3	$31\ell\log_2\ell+21\ell+6\sqrt{\ell}+4$	19
Extended-	II	$2\ell+5$	4	$31\ell\log_2\ell+21\ell+6\sqrt{\ell}+5$	20
Range	III	$2\ell+7$	5	$31\ell\log_2\ell+21\ell+6\sqrt{\ell}+7$	21

Table 8 Complexities of private exponentiation protocols and of Joint Random Number Bitwise-Sharing protocols used in them.

Protocol		Random number sharing		Private exponentiation	
		Comm.	Rounds	Comm.	Rounds
Ning and Xu [7]		$128\ell+24\sqrt{\ell}$	7	$157\ell+(20n+36)\sqrt{\ell}+10n+8$	24
Extended-	I	$16\ell+16$	4	$45\ell+(20n+12)\sqrt{\ell}+10n+24$	21
Range	II	$16\ell+24$	4	$45\ell+(20n+12)\sqrt{\ell}+10n+32$	21
Lightweight	I	$4\ell+8$	3	$33\ell+(20n+12)\sqrt{\ell}+10n+16$	20
Extended-	II	$4\ell+10$	4	$33\ell+(20n+12)\sqrt{\ell}+10n+18$	21
Range	III	$4\ell+14$	5	$33\ell+(20n+12)\sqrt{\ell}+10n+22$	22

6. Conclusion

Secure multi-party computation (MPC) is a promising tool for making use of personal or classified information while keeping it confidential, but one of the biggest problems with MPC is that it requires a vast amount of communication and thus a vast amount of processing time. Our analysis of existing MPC protocols revealed that the random number sharing protocol used by many of them is notably inefficient. The Joint Random Bitwise-Sharing protocol, which is used to generate random numbers in binary form, is particularly inefficient. By replacing truth values 1 and 0 used in this protocol by 0 and a non-zero value, we constructed more efficient protocols, commonly dubbed "Extended-Range I and II." Compared with the best of the existing Joint Random Bitwise-Sharing protocol, which requires $52\ell + 24\sqrt{\ell}$ multiplications and 7 rounds, the complexity is $8\ell + 8$ multipli-

cations and 4 rounds for Extended-Range I and $8\ell + 12$ multiplications and 4 rounds for Extended-Range II. Furthermore, by reducing the abort probability and thus making the previously necessary backup computation unnecessary, we constructed improved protocols, dubbed "Lightweight Extended-Range I, II, and III." The complexity is $2\ell + 4$ multiplications and 3 rounds for Lightweight Extended-Range I, $2\ell + 5$ multiplications and 4 rounds for Lightweight Extended-Range II, and $2\ell + 7$ multiplications and 5 rounds for Lightweight Extended-Range III. Using Lightweight Extended-Range II reduced the communication complexity for equality testing, comparison, interval testing, and bit-decomposition, all of which use a random number sharing protocol, by approximately 91, 79, 67, and 23%, respectively (for 32-bit data). We also reduced the communication complexity of private exponentiation by about 70% (for 32-bit data and five parties). Our protocols are fundamental to sharing random number $r \in \mathbb{Z}_p$ in binary form and can be applied to other higher level protocols.

References

[1] Bar-Ilan, J. and Beaver, D.: Non-Cryptographic Fault-Tolerant Computing in a Constant Number of Rounds of Interaction, *Proc. 8th Annual ACM Symposium on Principles of Distributed Computing*, pp.201–209, ACM (1989).

[2] Ben-Or, M., Goldwasser, S. and Wigderson, A.: Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation, *Proc. 20th Annual ACM Symposium on Theory of Computing*, pp.1–10, ACM (1988).

[3] Cramer, R., Damgård, I. and Ishai, Y.: Share Conversion, Pseudorandom Secret-Sharing and Applications to Secure Computation, *Proc. TCC 2005*, Kilian, J. (Ed.), LNCS, Vol.3378, pp.342–362, Springer (2005).

[4] Damgård, I., Fitzi, M., Kiltz, E., Nielsen, J. and Toft, T.: Unconditionally Secure Constant-Rounds Multi-party Computation for Equality, Comparison, Bits and Exponentiation, *Proc. TCC 2006*, Halevi, E. and Rabin, T. (Eds.), LNCS, Vol.3876, pp.285–304, Springer (2006).

[5] Gennaro, R., Rabin, M. and Rabin, T.: Simplified VSS and Fast-track Multiparty Computations with Applications to Threshold Cryptography, *Proc. 17th Annual ACM Symposium on Principles of Distributed Computing*, pp.101–111, ACM (1988).

[6] Ning, C. and Xu, Q.: Multiparty Computation for Modulo Reduction without Bit-Decomposition and A Generalization to Bit-Decomposition, *Proc. ASIACRYPT 2010*, Abe, M. (Ed.), LNCS, Vol.6477, pp.483–500, Springer (2010).

[7] Ning, C. and Xu, Q.: Constant-Rounds, Linear Multi-party Computation for Exponentiation and Modulo Reduction with Perfect Security, Cryptology ePrint Archive, Report 2011/069 (online), available from ⟨http://epring.iacr.org/⟩ (accessed 2011-09-29).

[8] Nishide, T. and Ohta, K.: Multiparty Computation for Interval, Equality, and Comparison Without Bit-Decomposition Protocol, *Proc. PKC 2007*, Okamoto, T. and Wang, X. (Eds.), LNCS, Vol.4450, pp.343–360, Springer (2007).

[9] SecureSCM: Security Analysis Technical Report D9.2, SecureSCM (online), available from ⟨http://www.securescm.org⟩ (accessed 2011-06-08).

[10] Shamir, A.: How to Share a Secret, *Comm. ACM*, Vol.22, No.11, pp.612–613 (1979).

[11] Toft, T.: Constant-Rounds, Almost-Linear Bit-Decomposition of Secret Shared Values, *Proc. CT-RSA 2009*, Fischlin, M. (Ed.), LNCS, Vol.5473, pp.357–371, Springer (2009).

Appendix

A.1 Variations of Proposed Protocols

We can easily construct a series of protocols based on the idea of the proposed ones. We show two types of variations of our protocols. The first type is variations of the Extended-Range protocol for $p = (10\ldots01\ldots10\ldots01)_2$. Let $p = (10\ldots0110\ldots01)_2$ and $p_{\ell-1} = p_{m+1} = p_m = p_0 = 1$ for simplicity. The output of $r < p$ is computed as follows, and its complexity is 3 multiplications and 3 rounds.

$$[r_{\ell-1}] \times \left\{ \left(\sum_{i=m+2}^{\ell-2} [r_i] \right) + [r_{m+1}] \times [r_m] \times \left(\sum_{i=1}^{m-1} [r_i] \right) \right\}$$

The second type is variations of the Lightweight Extended-Range protocol for $p = (1\ldots10\ldots01\ldots1)_2$. Let $p = (1\ldots1001\ldots1)_2$ and $p_{c+1} = p_c = 0$ for simplicity. The output of $r < p$ is computed as follows, and its complexity is 3 multiplications and 2 rounds.

$$[r^*] \left\{ \left(\sum_{i=c+2}^{\ell-1} [\overline{r_i}] \right) + [\overline{r_{c+1}}][\overline{r_c}] \right\} \times \left\{ \left(\sum_{i=c+2}^{\ell-1} [\overline{r_i}] \right) + [r_{c+1}] + [r_c] + \left(\sum_{i=0}^{c-1} [\overline{r_i}] \right) \right\}$$

These variations increase the number of applicable primes for our protocols and help in the selection of a suitable prime in practical use.

Naoto Kiribuchi was born in 1987. He received his B.E. and M.E. degrees from University of Electro-Communications in 2010 and 2012. He is currently with NTT Secure Platform Laboratories.

Ryo Kato was born in 1988. He received his B.E. and M.E. degrees from University of Electro-Communications in 2010 and 2012. He is a doctoral student in the Graduate School of Informatics, University of Electro-Communications.

Takashi Nishide received a B.S. degree from the University of Tokyo in 1997, an M.S. degree from University of Southern California in 2003, and a Dr.E. degree from University of Electro-Communications in 2008. From 1997 to 2009, he had worked at Hitachi Software Engineering Co., Ltd., developing security products. Since 2009, he has been an Assistant Professor in Kyushu University. His primary research is in the areas of cryptography and information security.

Tsukasa Endo received her B.E. and M.E. degrees from University of Electro-Communications, in 2007 and 2009. She joined the Computer Architecture & Security Systems Laboratory, Corporate Research and Development Center of Toshiba Corp., in 2009 and continued research on information security.

Hiroshi Yoshiura received his B.S. and D.Sc. from the University of Tokyo, Japan, in 1981 and 1997. He is currently a Professor in the Graduate School of Informatics, University of Electro-Communications. Before joining UEC, he had been at Systems Development Laboratory, Hitachi, Ltd. He has been engaged in research on information security and privacy and received the President's Technology Award from Hitachi in 2000, the Best Paper Award from Information Processing Society of Japan in 2005 and 2011, the Industrial Technology Award from the Institute of Systems, Control and Information Engineers in 2005, the Best Paper Award of IEEE International Conference on Intelligent Information Hiding and Multimedia Signal Processing in 2006, and the Best Paper Award from Japan Society of Security Management in 2011. He is a member of the IEICE, IPSJ, JSSM, ISCIE, and IEEE.