To appear in *IEEE Transactions on Dependable and Secure Computing*.

Cryptanalysis of a Privacy-Preserving Aggregation Protocol

Amit Datta and Marc Joye

Abstract—Privacy-preserving aggregation protocols allow an untrusted aggregator to evaluate certain statistics over a population of individuals without learning each individual's privately owned data. In this note, we show that a recent protocol for computing an aggregate sum due to Jung, Li, and Wan (IEEE Transactions on Dependable and Secure Computing, 2015) is universally breakable, that is, anyone is able to recover each individual's private data from the corresponding ciphertext. We also describe an alternate collusion attack against their companion product protocol.

Index Terms—Privacy, data aggregation, cryptanalysis.

BELOW we briefly review the two Jung-Li-Wan privacy-preserving aggregation protocols. We refer the reader to the original paper [1] for more details. We show that their sum protocol can be trivially broken to obtain the plaintext value from the ciphertext. We also point out a collusion attack.

1 DESCRIPTION

Suppose that there is a population of n users, denoted by $\{1, ..., n\}$, as well as a designated entity called the aggregator. In a privacy-preserving aggregation protocol, the goal of the aggregator is to compute the aggregate value

$$S = \sum_{i=1}^{n} x_i \qquad \text{[sum protocol]}$$

or

$$P = \prod_{i=1}^{n} x_i \qquad \text{[product protocol]}$$

in an oblivious way, that is, without having access to the private data x_i in the clear.

The Jung-Li-Wan protocols involve two prime-order groups, $\mathbb{G}_1$ and $\mathbb{G}_2$. Specifically, for two large primes p and q with $q \mid (p-1)$, select at random $h \in (\mathbb{Z}/p\mathbb{Z})^\times$ and set $g_1 = h^{(p-1)/q} \bmod p$, provided that $g_1 \neq 1$. Next set $g_2 = {g_1}^p \bmod p^2$. Groups $\mathbb{G}_1$ and $\mathbb{G}_2$ are defined as $\mathbb{G}_1 := \langle g_1 \rangle \subset (\mathbb{Z}/p\mathbb{Z})^\times$ and $\mathbb{G}_2 := \langle g_2 \rangle \subset (\mathbb{Z}/p^2\mathbb{Z})^\times$. It is readily verified that $\mathbb{G}_1$ and $\mathbb{G}_2$ are of order q.

This work was performed at Technicolor, Los Altos, CA, 94022, USA.

The protocols crucially rely on the simple observation that for any $r_1, \ldots, r_n \in \mathbb{Z}/q\mathbb{Z}$, one has

$$\sum_{i=1}^{n} (r_{i+1} - r_{i-1}) r_i = 0 \pmod{q} \tag{1}$$

where $r_{n+1} := r_1$ and $r_0 := r_n$.

1.1 Product Protocol

The product protocol works in group $\mathbb{G}_1$. Each user i, $1 \leq i \leq n$, chooses a random integer $r_i \in \mathbb{Z}/q\mathbb{Z}$ and evaluates

$$R_i = ({g_1}^{r_{i+1}} / {g_1}^{r_{i-1}})^{r_i} \bmod p\,,$$

where the quantity ${g_1}^{r_{i+1}} \bmod p$ (resp. ${g_1}^{r_{i-1}} \bmod p$) is received from user $i+1$ (resp. user $i-1$). Let $x_i \in \mathbb{Z}/p\mathbb{Z}$ denote the private data of user i. User i forms the corresponding ciphertext C_i as

$$C_i = x_i \cdot R_i \bmod p\ .$$

Upon collecting all the C_i's, the aggregator can then recover the product $P := \prod_{i=1}^{n} x_i \pmod{p}$ as

$$P = \prod_{i=1}^{n} C_i \pmod{p}\ .$$

The correctness of the protocol follows from Eq. (1) by remarking that $\prod_{i=1}^{n} R_i \equiv {g_1}^{\sum_{i=1}^{n}(r_{i+1}-r_{i-1})r_i} \equiv 1 \pmod{p}$.

1.2 Sum Protocol

The sum protocol goes in a similar way but in group $\mathbb{G}_2$. Namely, each user i evaluates

$$R_i = ({g_2}^{r_{i+1}} / {g_2}^{r_{i-1}})^{r_i} \bmod p^2$$

for a randomly chosen $r_i \in \mathbb{Z}/q\mathbb{Z}$. This value R_i is used by user i to get the encryption of her private data x_i. Specifically, the corresponding ciphertext C_i is given by

$$C_i = (1 + x_i\, p) \cdot R_i \bmod p^2\ . \tag{2}$$

From all the C_i's, the aggregator can then obtain the sum $S := \sum_{i=1}^{n} x_i \pmod{p}$ as

$$S = \frac{(\prod_{i=1}^{n} C_i \bmod p^2) - 1}{p} \pmod{p}\ .$$

Again the correctness of the protocol is easy to check. Indeed, we have $\prod_{i=1}^{n} C_i \equiv \prod_{i=1}^{n} [(1+x_i\, p) \cdot R_i] \equiv \prod_{i=1}^{n} (1 + x_i\, p) \cdot \prod_{i=1}^{n} R_i \equiv (1 + \sum_{i=1}^{n} x_i\, p) \cdot 1 \equiv 1 + S\, p \pmod{p^2}$ since $\prod_{i=1}^{n} R_i \equiv {g_2}^{\sum_{i=1}^{n}(r_{i+1}-r_{i-1})r_i} \equiv 1 \pmod{p^2}$ from Eq. (1).

2 CRYPTANALYSIS

As recalled in §1.2, if x_i is the private data of user i, its encryption is given by $C_i = (1 + x_i\, p) \cdot R_i \bmod p^2$ where $R_i = (g_2{}^{r_{i+1}} / g_2{}^{r_{i-1}})^{r_i} \bmod p^2$; cf. Eq. (2).

It is worth noting that R_i is an element of $\mathbb{G}_2$. Since the order q of $\mathbb{G}_2$ divides $p - 1$, it follows that

$$R_i{}^{p-1} \equiv 1 \pmod{p^2},$$

and consequently,

$$C_i{}^{p-1} \equiv (1 + x_i\, p)^{p-1} \equiv (1 - x_i\, p) \pmod{p^2} .$$

The private data of any user i (i.e., x_i) can therefore be recovered by anyone from the corresponding ciphertext C_i as

$$x_i = \frac{1 - C_i{}^{p-1} \bmod p^2}{p} \pmod{p} .$$

3 A COLLUSION ATTACK

The previous attack does not apply to the product protocol. We present an additional collusion attack that works against both the product protocol and the sum protocol. As we already showed in the previous section that the sum protocol is insecure, we describe the attack against the product protocol.

User i's randomizer is computed as

$$R_i = (g_1{}^{r_{i+1}} / g_1{}^{r_{i-1}})^{r_i} \bmod p$$

where r_i is a random integer generated by user i. The quantities $g_1{}^{r_{i+1}} \bmod p$ and $g_1{}^{r_{i-1}} \bmod p$ are received from users $i + 1$ and $i - 1$ respectively, as part of the protocol. User i also shares $g_1{}^{r_i} \bmod p$ with both users $i + 1$ and $i - 1$. Hence, if users $i + 1$ and $i - 1$ collude, then they can compute the randomizer of user i as

$$R_i = (g_1{}^{r_i})^{r_{i+1}} / (g_1{}^{r_i})^{r_{i-1}} \bmod p .$$

With knowledge of the randomizer R_i, the colluding adversaries can then recover the private value x_i from the ciphertext C_i of user i as $x_i = C_i / R_i \bmod p$.

Collusion attacks were already considered in the original paper. The authors propose in [1, Sect. 5.6] an advanced protocol aimed at defending against collusion attacks of so-called "rushing" adversaries. Their idea is to make the randomizer of each user i dependent on the random secrets of a larger number of different users. Specifically, for some parameter $k \geq 1$, they suggest to define the randomizer R_i of each user i as

$$R_i = (g_1{}^{r_{i+k+1}} / g_1{}^{r_{i-1}})^{r_{i+k} r_{i+k-1} \cdots r_i} \bmod p ,$$

which is evaluated through $k + 1$ rounds of exchange among users —note the basic protocols correspond to the case $k = 0$. For example, for $k = 1$, the suggested countermeasure goes as follows. In the first round, each user $i + 1$ respectively receives from users $i + 2$ and $i - 1$ the quantities $g_1{}^{r_{i+2}} \bmod p$ and $g_1{}^{r_{i-1}} \bmod p$, and computes $Y'_{i+1} = (g_1{}^{r_{i+2}} / g_1{}^{r_{i-1}})^{r_{i+1}} \bmod p$. Next, in the second round, each user i receives from user $i + 1$ the quantity Y'_{i+1} and computes her randomizer R_i as $R_i = Y'_{i+1}{}^{r_i} \bmod p$. This generalized protocol is unfortunately insecure. It is even less secure than the basic protocols as a *single* malicious user (as opposed to two in our collusion attack) can break it. Suppose that user $i + 1$ maliciously sets Y'_{i+1} as $Y'_{i+1} = g_1{}^{a} \bmod p$ for some random integer a. From the knowledge of a and $g_1{}^{r_i} \bmod p$, user $i + 1$ can then easily recover the randomizer of user i as $R_i = (g_1{}^{r_i})^a \bmod p$ and thereby user i's private value x_i as $x_i = C_i / R_i \bmod p$.

REFERENCES

[1] T. Jung, X.-Y. Li, and M. Wan, "Collusion-tolerable privacy-preserving sum and product calculation without secure channel," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 1, pp. 45–57, 2015.

Amit Datta is a PhD candidate at Carnegie Mellon University, Pittsburgh since Fall 2012 in the Department of Electrical and Computer Engineering. He received his Bachelor of Technology from the Indian Institute of Technology Kharagpur in 2012. His research interests include computer security, privacy, and cryptography. He has authored research papers in venues such as CT-RSA, PETS, CSF, and Indocrypt.

Marc Joye received his PhD degree in Applied Sciences (cryptography) from the Université Catholique de Louvain, Belgium, in 1997. In 1998 and 1999, he was a post-doctoral fellow of the National Science Council, Republic of China. From 1999 to 2006, he was with the Card Security Group, Gemplus (now Gemalto), France. Since August 2006, he has been with Technicolor (formerly Thomson R&D), France & USA. His research interests include cryptography, privacy, computer security, computational number theory, and smart-card implementations. He is author and co-author of 130+ scientific papers and holds several patents. He served in numerous program committees and was program chair for CT-RSA 2003, CHES 2004, ACM-DRM 2008, FDTC 2010, ACM-DRM 2010, Pairing 2010, InfoSecHiComNet 2011, and CARDIS 2014. He is a member of the IACR and co-founder of the UCL Crypto Group.