November 2022

SECURITY MULTIPARTY ANONYMIZATION FOR SECURITY ANALYSIS OF DISTRIBUTED DEVICES

HP INC

Security Multiparty Anonymization for Security Analysis of Distributed Devices

Abstract. *A method for secure multiparty anonymized analytics using a federated schema is disclosed. The method involves a federation mechanism between the involved parties and a reversible pseudo anonymization threshold protocol, avoiding single point of failure attacks.*

Analytics should keep data private, but for security reasons, data needs to be deanonymized after a security incident. A centralized schema can be adopted but the devices can be compromised. We propose a federated solution, where sensitive data cannot be recovered without a quorum of the parties.

Given a set of *n* devices, each of them with the ability of sending sensitive data, and some of them capable of encrypting them using a cryptographic method. There are n **workers** in the system, and the role of the **worker** is to: (i) receive information, including keying material (a public key and a share of a private key), from the master; (ii) generate and encrypt data, (iii) send encrypted data to the master; (iv) participate in collaborative decryption if required. A **master** is selected from the set of workers. The master may change over time. The role of the master it to generate and distribute keying material and to coordinate the workers. We refer to the set of workers and the master as the 'federation'. In addition to the federation, two other elements established: (i) a trusted party (the system administrator) that set up the federation in the first place, and (ii) a third-party agent whose security triggers initiate the deanonymization process for forensics.

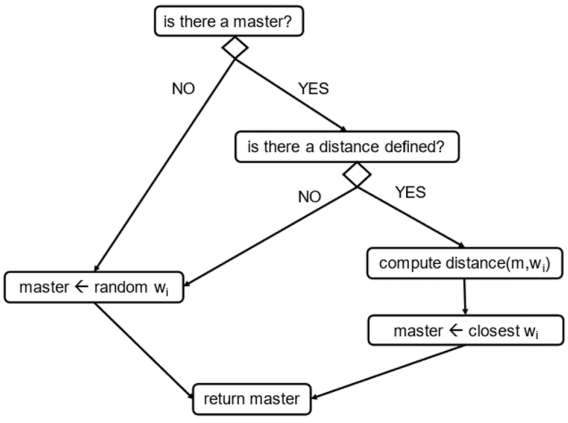

Figure 1. Diagram showing master selection

The system consists of the following stages:

Stage 1: Setup. First, a master must be assigned from the set of workers. To do this, we follow the following procedure. Let there be n workers devices. Let *c* be the number of devices that can be compromised at the same time, *m*, the number of devices in the federation, w_i a worker device, and *d* the distance computed between the devices. This distance *d* can be established *ad hoc* using different metrics that represent distance withing the fleet (e.g., computation power, network speed, network latency, similar configuration, trust etc.).

If this is the first time the federation is used, a random w_i is selected as manager. Once the master has been selected, the master must generate the keying material:

1. *The master generates an asymmetric encryption key pair, (pk, sk). The private decryption key sk is then distributed according to a (t, n)- threshold secret sharing scheme chosen by the administrator, resulting in n shares of the pk, (s_1, s_2, …, s_n).*
2. *The master securely sends worker w_i the share s_i and the public key pk.*
3. *The master deletes the private key sk.*

To reset the configuration of the federation, the produce to follow is the same.

Stage 2: Data anonymization. When a worker generates some data that must be encrypted, they use the public encryption key pk, received during stage 1, to encrypt the data: *encrypts (data, pk) = c.* Note that some of the data, or all the data can be anonymized depending on the requirements on the scenario. There are two options for where the worker stores the data depending on the configuration and the workers capability.

1. If all the ciphertexts are required to be stored in a central location, the worker sends the ciphertext to this central location (possibly via the master).
2. If the worker does not require a centralized storage and encryption, then the worker may store the data itself.

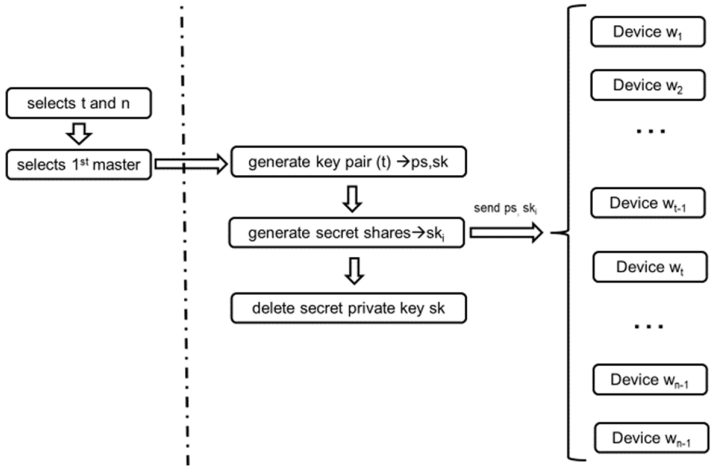

Figure 2. Diagram showing setup and data anonymization stages.

Stage 3: Data de-anonymization. (Figure 3). The master receives an outside trigger. There are two types of deanonymization process can be triggered: (i) a distributed deanonymisation, in which t of the n workers must collaborate, or (ii) a centralized deanonymisation executed by the master.

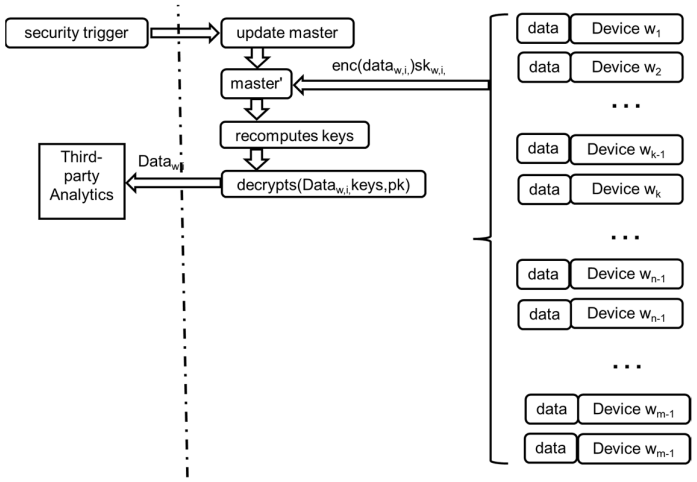

Figure 3. Diagram showing the centralised de-anonymisation process.

In a centralized (see Figure 3), the workers cannot participate in a collaborative protocol. Instead, each worker sends their share of the private key to the master. Then, the master retrieves the ciphertext (from the central location, of from a worker, for example). Finally, once the master has received enough shares (at least t), the master can decrypt the ciphertext to recover the plaintext data.

In a distributed collaborative approach (shown in Figure 4), when some data needs to be decrypted, the workers retrieve the previously stored ciphertext. Each worker then uses their share of the decryption key, s_i, and participates in a collaborative decryption protocol along with at least t-1 other workers to retrieve the plaintext data. Note that during this process, no worker reveals it share si of the private key, and the decryption key never exists on any one device. Finally, all workers send the plaintext data to the third-party to proceed the analysis.

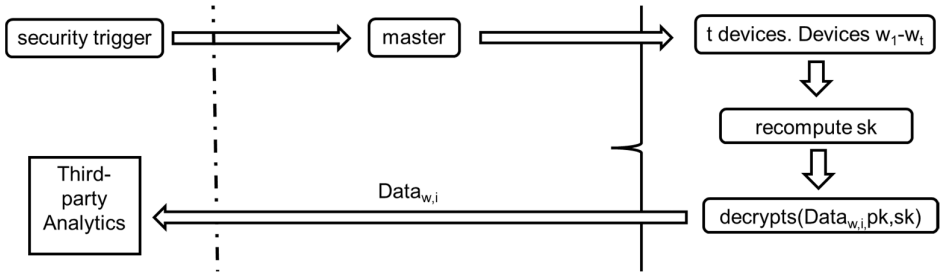

Figure 4. Diagram showing the decentralized de-anonymization process.

Stage 4. Federation management. Additional procedures are needed to deal with addition and subtraction of new worker devices.

When a new device enters the new device communicates with the current master. Then, the master recomputes the federation (as shown in Figure 1) by initiating the set-up procedure, but this time including the new device (so n -> n+1). This will result in a new master, a new public key, and new shares for the workers. Alternatively, t devices could send their shares to the master, the master would recover the private decryption key sk from these shares, then generate a new share for the new device (for example, if Shamir's threshold scheme is used, the master could compute the function f from the shares then compute f(n+1) as the new share for the new worker device). This solution means the public key would remain unchanged, which may be useful in settings where the master cannot necessarily connect with all n workers to give them a new share/public key.

In the case of a worker device leaving the federation, there are two different cases. First, if a device leaves the federation and other members in the federation know the device has left (for example, if the worker is revoked from the group by the system administrator, or is re-allocated to a different federation), the master is informed, then the master recomputes the federation (by executing the setup procedure, resulting in a new master and new keys and shares), excluding the worked device that has just left/been revoked. When a device fails, the problem is when it is detected by the federation. Without adding other information, it may be detected by the master during a deanonymization. Being n' the new number of remaining devices and t the required threshold for the protocol. If $t \leq n'$, then deanonymization procedure could be performed and n should be updated. In any case, to limit the possibility of an attack the federation is recomputed at discretion of the administrator. If $t < n'$¸ then the deanonymization is not possible and a new federation needs always to be recomputed.

To sum up. this system provides a method for devices to anonymise data, and for the data to be collaboratively de-anonymised for forensics if required.

Disclosed by Igor Santos, Daniel Ellam, Stuart Lees, Adrian Baldwin, Nelson L Chang and Thalia May Laing, HP Inc.