SECURE MULTI-PARTY CONSENSUS GOSSIP ALGORITHMS

Riccardo Lazzeretti[†] *Steven Horn*[⋆] *Paolo Braca*[⋆] *Peter Willett*[‡]

[†] University of Siena, Siena, Italy, Email: lazzeretti@diism.unisi.it
[⋆] NATO STO CMRE, La Spezia, Italy, Email: {horn/braca}@cmre.nato.int
[‡] University of Connecticut, Storrs CT, Email: willett@engr.uconn.edu.

ABSTRACT

Information fusion is the keystone of many surveillance systems, in which the security of the information is a crucial aspect. This paper proposes a method to fuse information exchanging only encrypted data, through a secure extension of the popular consensus gossip algorithm using secure multi-party computation methodology. Sensor entities exchange only encrypted information and never have direct access to the data while iteratively reaching consensus. The agents do not have access to the final value and can just retrieve partial information, for instance a binary decision. An innovative implementation of the consensus algorithm in the encrypted domain is proposed and analyzed.

Index Terms— Consensus algorithms; detection; secure multi-party computation; information fusion; encryption

1. MOTIVATION AND RELATED WORKS

In recent years, information fusion has received significant attention for both military and nonmilitary applications. Data fusion techniques combine data from multiple sources, as well as related information from associated databases, to achieve improved awareness and more specific inferences than could be achieved by the use of a single terminal alone [1]. In the modern paradigms of information fusion, the agents (sensors, or nodes) are organized in a network, often referred as wireless sensor networks (WSNs) [2]. Special attention is dedicated to the integrity of information since even if the communication protocols in the network can be considered secure (i.e. transport layer security), due to the unique challenges in sensor networks, standard security techniques used in traditional networks cannot be applied directly [3]. For instance in [4, 5] the case of a group of agents, reprogrammed by an opponent, to work against the network is considered.

This paper considers the case in which the agents have an interest in cooperation but at the same time do not want to reveal their own information for security reasons and privacy preservation (e.g. classified information, protection of sensor characteristics, etc.). Interaction is achieved by exchanging *encrypted* information between agents. The information fusion mechanism is then executed in the encrypted domain [6]. To clarify this situation, assume that a pair of agents, say i and j, want to make a binary decision $\{\mathcal{H}_0, \mathcal{H}_1\}$ based on their data x_i and x_j, respectively (see Fig. 1). If these observations are conditionally independent, then the optimal decision statistic is the summation of the log-likelihood ratios (LLRs) $\mathcal{L}(x_i) + \mathcal{L}(x_j)$, where $\mathcal{L}(x) = \log\left(f(x;\mathcal{H}_1)/f(x;\mathcal{H}_0)\right)$ and $f(x;\mathcal{H}_{0,1})$ is the distribution under $\mathcal{H}_{0,1}$. However, the agents i and j do not want to reveal their data, but can exchange the encryption $[\![\mathcal{L}(x_{i,j})]\!]$ of their data. Is it possible to obtain the optimal decision by agents exchanging only encrypted data? A solution to this intriguing problem for a class of WSNs by using the popular iterative gossip consensus protocol [7, 8] in the encrypted domain is here proposed.

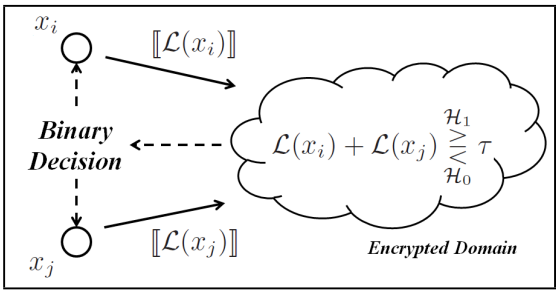

Fig. 1. Conceptual scheme of cooperation between agents i and j to solve a binary decision testing problem in a privacy preserving setting. Agents can communicate with each other and exchange only encrypted data, for instance an encrypted version of the LLR $[\![\mathcal{L}(x_{i,j})]\!]$, however at the same time they want to have access to the binary decision based on both the observations x_i and x_j.

The consensus paradigm fits well with the nature of the WSNs, where agents may enter or leave the network dynamically, resulting in unpredictable changes in network size and topology. Agents may also disappear permanently either due to damage or drained batteries, etc., or temporarily due to topology, traffic, and communication conditions. This makes it necessary for distributed signal processing algorithms to be robust to any changes in network topology. Among the several system architectures, communication strategies, and cooperative procedures proposed for sensor networks, of particular interest is the fully flat architecture where the remote units sense the environment and collect data, but, due to the lack of a fusion center, they are also programmed to run a consensus procedure aimed at corroborating the local measurements with observations made by neighboring agents. The process of data exchange updates the locally computed statistics and (asymptotically) leads to the agreement about a common value shared by all the agents, that represents the final statistic. Useful entry points to the topical literature on consensus include [7, 8, 9, 10, 11, 12, 13, 14].

As well as in other privacy preserving applications such as data mining [15], biometric matching [16], recommendation systems [17], biomedical analysis [18], etc., the consensus protocol can be executed in the encrypted domain [6] by using secure Multi-Party Computation (MPC) protocols to reach the consensus about a common value while each agent has access *only* to its inputs and the final

S. Horn and P. Braca have been funded by the NATO Allied Command Transformation (NATO-ACT) under the project Maritime Situational Awareness.

decision, obtained by evaluating the protocol on encrypted statistics.

To the best of our knowledge, this is a completely novel work where MPC is applied to a fully decentralized WSN. Previous works in privacy preserving data fusion were addressed in [19, 20], where the authors propose some techniques, based on additive blinding or secret sharing, to estimate the position of one or more targets by computing the mean of the measurements of multiple sensors which are connected through a network where a path exists starting and finishing at each agent and passing through all the agents. By comparison, our solution fuses the data through a privacy preserving implementation of the consensus algorithm and also allows data fusion in networks with low connectivity degrees, and in dynamic networks.

2. SECURE MULTI-PARTY COMPUTATION

This section presents an overview of the most used techniques for MPC, namely Homomorphic Encryption (HE) and Garbled Circuits (GC). Throughout the paper, the semi-honest security model is adopted, where the parties involved follow the protocol as prescribed but try to learn as much as possible from the messages exchanged and their inputs.

Homomorphic Encryption. With a semantically secure, additively homomorphic, asymmetric encryption scheme, having dinstinct public and private encryption keys *PuK* and *PrK*, it is possible to compute the encryption of $[\![a+b]\!]$, by the encryptions $[\![a]\!]$ and $[\![b]\!]$ and the product between an encrypted number and a public factor. A commonly used additively homomorphic cryptosystem is the Paillier cryptosystem [21] which has plaintext space $\mathbb{Z}_n$ and ciphertext space $\mathbb{Z}^*_{n^2}$, where n is a T-bit RSA modulus and a ciphertext is represented with $2T$ bits (commonly $T = 1024$ bits). The Paillier cryptosystem permits the evaluation of the encryption of the sum of two values through the product of the corresponding ciphertexts $[\![a+b]\!] = [\![a]\!] \cdot [\![b]\!]$ and the product between two values, one of them public, through exponentiation: $[\![ab]\!] = [\![a]\!]^b$. Moreover, other functions can be computed through interactive protocols based on additive blinding, some of them having efficient implementation (such as product between two ciphertexts), others having high complexity (comparison, division, etc.). The computational complexity of HE solution is related to the number of exponentiations performed by the protocol (encryptions and decryptions have a complexity similar to the exponentiations, while products involving ciphertexts have negligible complexity respect to the other operations). The communication complexity is related to the number of ciphertexts transmitted. Recently, starting from the work of Gentry [22], Fully Homomorphic Encryption (FHE) schemes have been proposed. Unfortunately, these cryptosystems are not yet efficient due to the bit-length of ciphertexts (each one encrypting a single bit) and public key.

Garbled Circuit. First proposed in the seminal work of Yao [23, 24], Garbled Circuit (GC) protocols permit the evaluation of any boolean circuit with communication and computational overhead depending on input bit length and circuit size. GC, as outlined in [25, 26] and shown in Fig. 2, is composed of three sub-routines: circuit garbling, data exchange and evaluation.

First, a party garbles the gates and the wires composing the circuit. Then the circuit, together with the secrets relative to the inputs, is transmitted to another party that evaluates the garbled circuit to obtain the final result. It is important to underline that the complexity of the tool is related to the number of non-XOR gates, each one having a garbled table associated (of size $3t$ bits, where t is a security parameter, usually $t = 80$ bits) that is transmitted from the garbler to the evaluator and whose garbling and evaluation are performed by using respectively 3 and 1 hash functions, while XOR gates have negligible computational and communication complexity. The secrets associated to the garbler's input bits (t bits each) are transmitted from the garbler to the evaluator, after having associated them to the real inputs, while the evaluator secret transmission involves Oblivious Transfer (OT) [27] that associates the input bits to secrets chosen by the circuit garbler. Considering that OT can be precomputed [28], many OT's can be evaluated off-line on random values (regardless of the actual values used during the circuit evaluation) and resulting in a lower on-line communication complexity, only $2t$ bits for each input bit. The performance of GC is related to the number of gates composing the circuit. Some operations have low performance cost (additions, comparisons, etc.), while others are quite expensive (products, divisions).

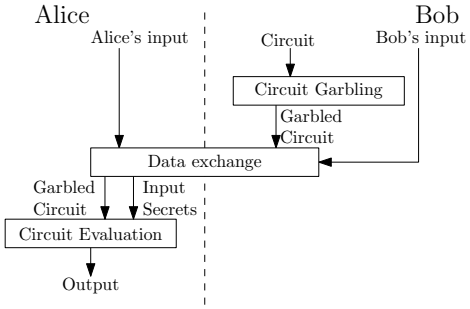

Fig. 2. Garbled Circuits scheme.

Hybrid protocols. The use of hybrid protocols, as described in [26], permits efficient evaluation of complex functions for which full-HE or full-GC solutions would be inefficient (or even impossible). Due to the different representation of the data for GC and HE, conversion from a homomorphic ciphertext $[\![x]\!]$ to garbled secrets (or vice versa) must be performed by using additive blinding: given that the plain x can be correctly represented with ℓ bits, a random value of $\ell + t$ bits is chosen and then added to $[\![x]\!]$ by using homomorphic property, then it is transmitted to the private key owner that decrypts it and uses the least significant ℓ bits in a GC, where the other party inputs the least significant ℓ bits of the random value, used to remove the obfuscation through a difference[1], before the circuit implementing the given functionality is evaluated. Similarly, the output of GC computation can be translated into a ciphertext.

Data representation. MPC protocols are different than typical applications in that they need to work with integer numbers. The principal reason for this is that homomorphic cryptosystems do not support encryption of floating point numbers. Even if representation of floating point numbers can be done in GC, the complexity of the protocols will increase. If higher precision is required, the input values are usually quantized, multiplying them by a factor q (typically a power of 2) before rounding, i.e. each x_i is represented by using an integer number X_i obtained as $\lfloor x_i \times q \rfloor$ and can be represented with $\ell = \lceil \log_2 \max\{X\} \rceil + \lceil \log_2 q \rceil$ bits. Is it also possible to approximate a value to a close rational number and then represent each value X as a ratio n/d between a numerator n and a denominator d. This also permits the avoidance of computing expensive divisions in the encrypted domain.

Security. The security of hybrid protocols in the semi-honest setting, as proven in [26], is guaranteed by the security of GC scheme and HE cryptosystems and the use of additive blinding in the inter-

[1] Even if all the $\ell+t$ bits are used as input, the subtraction already provides the correct results after that ℓ bits are processed, making the others useless.

faces. The security of the GC parts is guaranteed by the security of the GC primitive, while the parts implemented by using HE are secure, given the IND-CPA property of the encryption protocol used.

3. PROTOCOL IMPLEMENTATION

The network is formally described by an undirected graph, whose vertices are the agents (or nodes) and edges are the available communication links. The set of nodes neighbouring the i^{th} node is denoted by $\mathcal{N}_i \subseteq [1, 2, \ldots, N]$, the node i can communicate solely with nodes belonging to $\mathcal{N}_i$. The adjacency matrix associated to the network graph is denoted by A, with $\{A\}_{ij} = 1$ if $j \in \mathcal{N}_i$, otherwise $\{A\}_{ij} = 0$. Since $j \in \mathcal{N}_i$ if and only if $i \in \mathcal{N}_j$, A is symmetric, i.e., $A = A^T$. Using the adjacency matrix, a random averaging consensus matrix is defined $W(k)$ at time step k, so that the nodes update their state $y(k-1)$ according to the iterative rule

$$y(k) = W(k)y(k-1), \tag{1}$$

where the initial state is given by the local decision statistic $y(0) = [\mathcal{L}(x_1), \mathcal{L}(x_2), \ldots, \mathcal{L}(x_N)]^T$, given by the LLR $\mathcal{L}(x)$ between $f(x; \mathcal{H}_1)$ and $f(x; \mathcal{H}_0)$, the distributions under $\mathcal{H}_1$ and $\mathcal{H}_0$, respectively. The consensus procedure has interesting properties, in particular, under mild conditions (e.g. connectivity of the network graph) the convergence is guaranteed to the average of the initial values $\sum_{i=1}^{N} \mathcal{L}(x_i)/N \leftarrow y_i(k), \forall i = 1, \ldots, N$, see details in [7, 9, 11]. Each node is interested in computing the binary statistical decision $\mathcal{D}_i(k) \in \{\mathcal{H}_0, \mathcal{H}_1\}$ given by the test $y_i(k) \gtrless_{\mathcal{H}_0}^{\mathcal{H}_1} \tau$.

For simplicity this work focuses on the *randomized gossip algorithm* [7, 8] where at each consensus step k a pair of adjacent nodes (able to communicate), say node i and j, is randomly selected according to the network graph. These nodes exchange their information and update their states at time k by the averaging rule [7]

$$y_i(k) = y_j(k) = \frac{y_i(k-1) + y_j(k-1)}{2}, \tag{2}$$

while the other nodes hold their previous value $y_l(k) = y_l(k-1), \forall l \neq i, j$. Since the MPC implementation of division is expensive, it is avoided by representing each node state as a ratio between a numerator n_i and a denominator d_i. Hence, given $y_i(k-1) = n_i(k-1)/d_i(k-1)$ and $y_j(k-1) = n_j(k-1)/d_j(k-1)$, computed the least common multiplier $lcm(k)$ between $d_i(k-1)$ and $d_j(k-1)$, at the end of the step the nodes obtain

$$\begin{aligned} n_i(k) &= n_j(k) \\ &= n_i(k-1)\frac{lcm(k)}{d_i(k-1)} + n_j(k-1)\frac{lcm(k)}{d_j(k-1)}, \\ d_i(k) &= d_j(k) = 2\, lcm(k). \end{aligned} \tag{3}$$

Considering that while the numerator carries information related to the inputs, the denominator only depends on the operations performed and therefore it is not necessary to keep it secret. Note that since $d_i(0) = 1\ \forall i$, $lcm(0) = 1$ and $d_i(1) = 2$. During the computation one can easily infer that $d_i(k-1)$ are powers of 2, $\forall i, k$ and hence each $lcm(k)$ can be computed as $\max\{d_i(k-1), d_j(k-1)\}$.

In the final step K, each node evaluates the comparison with its threshold, with the help of an adjacent node.

A secure implementation of the consensus algorithm that preserves the secrecy of data and decision statistics (1), allowing the sensors to retrieve the decision $\mathcal{D}_i(k)$, is proposed (see also Fig. 3). Considered first is the case that the connections among the nodes, and hence A, is not time variant. This implementation can then later be extended to a time variant network topology. In this protocol, the nodes do not have access to any plaintext numerators and each node owns a different private and public key pair, hence the encryption of m obtained by using the public key PuK_i of the node i is denoted by $[\![m]\!]_i$.

Step k. Each agent keeps the data encrypted with the keys of all the other agents, i.e. at step k agent i owns $n_i(k-1)$ encryptions with the keys of all its neighbors, hence it has $[\![n_i(k-1)]\!]_l$ available for each $a \in \mathcal{N}_i$.

When the agent i cooperates with j, it selects $[\![n_i(k-1)]\!]_j$ while j selects $[\![n_j(k-1)]\!]_i$. Agent i adds a random value $r_i(k)$ to $[\![n_i(k-1)]\!]_j$ and sends it to j together with $[\![-r_i(k)]\!]_i$ and $d_i(k-1)$. Agent j first decrypts $[\![n_i(k-1) + r_i(k)]\!]_j$ by using its private key PrK_j and then encrypts it again by using the public key of i. Then agent j, after obtaining $lcm(k)$, computes

$$\begin{aligned} [\![n_j(k)]\!]_i &= ([\![n_i(k-1) + r_i(k)]\!]_i [\![-r_i(k)]\!]_i)^{\frac{lcm(k)}{d_i(k-1)}} \\ &\quad \times [\![n_j(k-1)]\!]_i^{\frac{lcm(k)}{d_j(k-1)}} \\ &= [\![n_i(k-1)\frac{lcm(k)}{d_i(k-1)} + n_j(k-1)\frac{lcm(k)}{d_j(k-1)}]\!]_i, \\ d_j(k) &= 2\, lcm(k). \end{aligned} \tag{4}$$

In parallel agents i and j follow the protocol inverting the roles, so that also i obtains $[\![n_i(k)]\!]_j$ and $d_i(k)$. At this point the agent i needs to compute $[\![n_i(k)]\!]_a$, $\forall a \in \mathcal{N}_i/\{j\}$, while j needs to compute $[\![n_j(k)]\!]_b$ $\forall b \in \mathcal{N}_j/\{i\}$. Considering that $n_i(k) = n_j(k)$, the agent j adds a second random value $s_j(k)$ to $[\![n_j(k)]\!]_i$ and sends it to i together with $[\![-s_j(k)]\!]_a$. Finally agent i decrypts $n_j(k) + s_j(k)$ and computes the encryption of the state under all the other adjacent agents' public keys as $[\![n_i(k)]\!]_a = [\![n_j(k) + s_j(k)]\!]_a [\![-s_j(k)]\!]_a$. Again, j obtains all the $[\![n_j(k)]\!]_b$ by following the same procedure in parallel with inverted roles. The state encrypted with all the keys of the adjacent nodes is obtained at the end of step k. Both agents i and j can later cooperate with any other node. In the case of dynamic networks, it is required that the node i obtains $n_i(k)$ encrypted with the public keys of all the agents, maybe discarding only the nodes that, for different reasons, agent i will never communicate with.

Final step. The final goal of an agent i is to discover if the final value obtained by the consensus protocol is greater than a given threshold τ. To obtain the final result it is sufficient that i cooperates with any adjacent node j to evaluate $n_i(K)/d_i(K) \gtrless \tau$. Since $d_i(K)$ is known by agent i, the final result can be obtained with the following protocol: first of all it adds a random value $r(K)$ to $[\![n_i(K)]\!]_j$ and transmits $[\![n_i(K) + r(K)]\!]_j$ to j. Finally they evaluate together a GC computing the circuit $(n_i(K) + r(K)) - (r(K)) \gtrless \tau\, d_i(k-1)$ where $n_i(k-1) + r(K)$ is the input from j (acting as garbler), while $r(K)$ and $\tau\, d_i(K)$ (computed in the plain domain) are inputs from i (acting as evaluator). To design the circuit, the bitlength of $n(K)$ has to be determined. Given the bitlength ℓ of $n_i(0)$, one can easily observe that $lcm_i(1) = 1$, $d_i(1) = 2$ and hence $\ell + 1$ bits are needed to represent $n_i(1)$. Iterating the computation one obtains that at step k, $lcm(k)$ is a power of 2 lower than 2^{k-1} and $n_i(k)$ can be represented by using a number of bits equal to the bitlength of $n_i(k-1)$ plus k. It is concluded that $n_i(K)$ would quickly grow to $\ell + \frac{K(K+1)}{2} + K$ bits. The numerator bitsize can be reduced by dividing it (and also the denominator) by a given factor when the bitlength exceeds a given $\ell + \ell_1$ with one of the protocols described in [29, 30]. Considering the above numerator reduction, $r(K)$ is chosen in $\mathbb{Z}_{2^{\ell+\ell_1+\iota}}$, as described in Section 2.

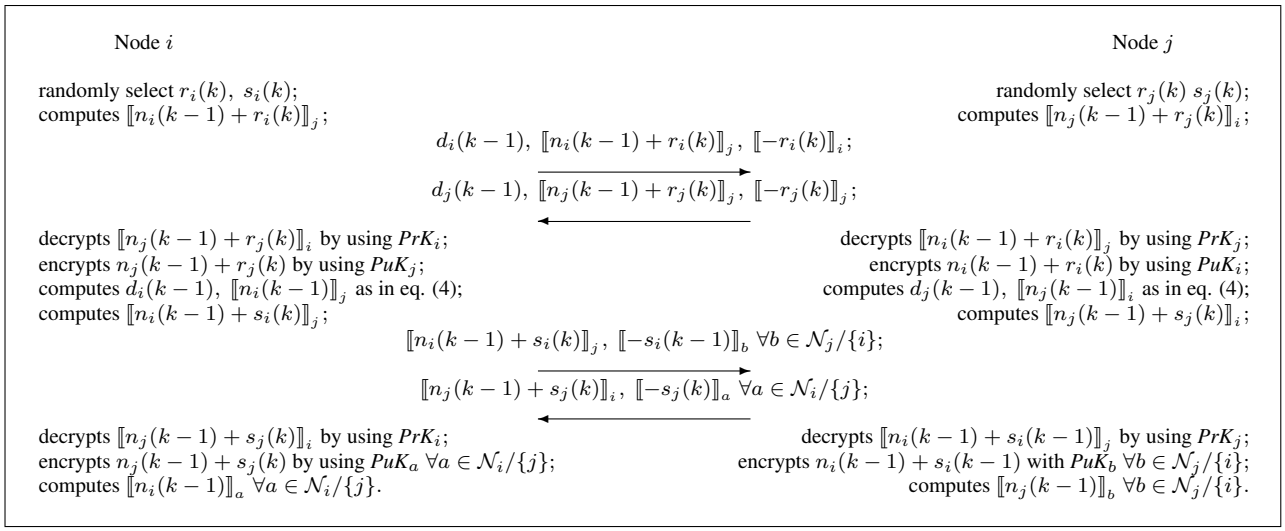

Fig. 3. Step k of the secure consensus gossip protocol.

4. ANALYSIS

The computational and communication complexity of the protocol for non-dynamic networks is analyzed. As stated in Section 2, the computational complexity of the HE parts depends on the number of exponentiations performed while the communication complexity depends on the number of ciphertexts transmitted (whose size is $2T$ bits each). For the GC part, the computational complexity depends on the number of hash functions evaluated (4 for each non-XOR gate), while the communication complexity depends on the bitlength of evaluator inputs ($2t$ bits transmitted for each input bit), the bitlength of garbler inputs (t bits transmitted for each input bit) and the number of non-XOR gates composing the circuit ($3t$ bits transmitted for each non-XOR gate). The OT can be performed before the protocol starts, or while two adjacent nodes haven't updated their values, and therefore the complexity of this "offline" calculation is not considered here.

In a generic step k of the consensus protocol, the agent i encrypts $r_i(k)$ with the key PuK_j and $-r_i(k)$ with the key PuK_i; transmits 2 ciphertexts to j and receives 2 ciphertexts from it, then it decrypts one of the received ciphertexts by using PrK_i and encrypts it again with the key PuK_j; computes $[\![n_i(k+1)]\!]_j$ by performing 2 exponentiations; encrypts $s_i(k)$ to obfuscate $n_i(k+1)$ and moreover $-s_i(k)$ is encrypted with the public keys of the adjacent nodes of j, except PuK_i ($|\mathcal{N}_j|-1$ encryptions, where $|\cdot|$ denotes the cardinality of the set); transmits $|\mathcal{N}_j|$ ciphertexts to j while receive $|\mathcal{N}_i|$ ciphertexts from j; after receiving $[\![n_j(k+1)+s_j(k)]\!]_i$, node i decrypts it and encrypts again with the other $|\mathcal{N}_i|-1$ keys before removing the obfuscation.

In total the agent i performs $|\mathcal{N}_i|+|\mathcal{N}_j|+4$ exponentiations. Also the agent j computes in parallel $|\mathcal{N}_i|+|\mathcal{N}_j|+4$ exponentiations. Hence in each step $2(|\mathcal{N}_i|+|\mathcal{N}_j|)+8$ exponentiations are computed. From a communication point of view, the two agents involved in the computation in step k transmit $|\mathcal{N}_i|+|\mathcal{N}_j|+2$ ciphertexts.

In the final step the following operations are performed by each agent: after choosing an adjacent agent j, the agent i performs an encryption for the value obfuscation, transmits a ciphertext to the agent j that decrypts a value and finally they evaluate together a GC. The evaluator (agent i) has 2 $(\ell+\ell_1)$ bit-long inputs in the worst case, hence their association to garbled secrets through OT requires the transmission of $2(\ell+\ell_1)(2t)$ bits, while the garbler (node j) has a $(\ell+\ell_1)$ bit-long input, hence the transmission of the secrets associated to its input result in $(\ell+\ell_1)t$ bits. The circuit is composed by a subtracter and a comparison circuit, both of them having $\ell+\ell_1$ non-XOR gates, hence $2(\ell+\ell_1)3t$ bits are transmitted for the circuit and $4\times 2(\ell+\ell_1)$ hash functions are evaluated in total.

Step	Computational		Communication
	Expo	Hash	bits
k	$2(\lvert\mathcal{N}_i\rvert+\lvert\mathcal{N}_j\rvert)+8$	0	$(\lvert\mathcal{N}_i\rvert+\lvert\mathcal{N}_j\rvert+2)2T$
Final	$2N$	$8(\ell+\ell_1)N$	$2NT+11(\ell+\ell_1)Nt$

Table 1. Complexity of the protocol for all the nodes in each step.

Recalling that the operations involved in the consensus protocol are repeated K times while the final step is performed by all the N nodes of the network, the complexities of the protocol are summarized in Table 1. The choice of K can be based on the concept of ϵ-averaging time [8], i.e. the earliest gossip time in which the state vector $y(k)$ is ϵ away from the normalized true average with probability greater than $1-\epsilon$. A sufficiently small ϵ, which guarantees that all agents take the same decision with high probability, requires setting $K=\mathcal{O}\left(\frac{\log\epsilon^{-1}}{1-\lambda_2(\mathbb{E}W)}\right)$, where $\mathbb{E}$ is the expected value operator and $\lambda_2(\mathbb{E}W)$ is the second largest eigenvalue of $\mathbb{E}W$ (<1 if the network graph is connected).

5. CONCLUSIONS

A mechanism to fuse information working in the encrypted domain through a secure extension of the consensus algorithm based on the MPC methodology is proposed in this paper. The developed protocol is based on the gossip algorithms, in which a pair of agents participate in data exchange in each time frame. Future work includes the implementation and experimental validation of this protocol for application in WSNs and other potential areas, generalization to include the case of time-varying network topologies and distributed adaptive diffusion algorithms, and considering a more robust protocol which is resilient to the case in which some agents may be dishonest.

6. REFERENCES

[1] D.L. Hall and J. Llinas, "An introduction to multisensor data fusion," *Proc. IEEE*, vol. 85, no. 1, pp. 6–23, 1997.

[2] J. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Commun. Mag.*, vol. 40, pp. 102–114, Aug. 2002.

[3] A. Perrig, J. Stankovic, and D. Wagner, "Security in wireless sensor networks," *Comm. ACM*, vol. 47, no. 6, pp. 53–57, 2004.

[4] S. Marano, V. Matta, and L. Tong, "Distributed detection in the presence of Byzantine attacks," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 16–29, 2009.

[5] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Trans. Program. Lang. and Syst.*, vol. 4, pp. 382–401, July 1982.

[6] R. Lagendijk, Z. Erkin, and M. Barni, "Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation," *IEEE Signal Process. Mag.*, vol. 30, no. 1, pp. 82–105, 2013.

[7] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, June 2006.

[8] A.G. Dimakis, S. Kar, J.M.F. Moura, M.G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, 2010.

[9] R. Olfati-Saber, A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[10] P. Braca, S. Marano, and V. Matta, "Enforcing consensus while monitoring the environment in wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3375–3380, 2008.

[11] P. Braca, S. Marano, V. Matta, and P. Willett, "Asymptotic optimality of running consensus in testing statistical hypotheses," *IEEE Trans. Signal Process.*, vol. 58, no. 2, pp. 814–825, 2010.

[12] P. Braca, S. Marano, V. Matta, and P. Willett, "Consensus-based Page's test in sensor networks," *Sig. Proc.*, vol. 91, no. 4, pp. 919–930, Apr. 2011.

[13] S. Kar and J. Moura, "Consensus + innovations distributed inference over networks: cooperation and sensing in networked systems," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 99–109, 2013.

[14] G. Battistelli, L. Chisci, C. Fantacci, A. Farina, and A. Graziano, "Consensus CPHD filter for distributed multitarget tracking," *IEEE J. Sel. Topics Signal Process.*, 2013.

[15] Y. Lindell and B. Pinkas, "Privacy preserving data mining," *Journ. crypt.*, vol. 15, no. 3, pp. 177–206, 2002.

[16] Y. Luo, S. Samson, T. Pignata, R. Lazzeretti, and M. Barni, "An Efficient Protocol for Private Iris-Code Matching by Means of Garbled Circuits," in *Int. Conf. Im. Proc. (ICIP)*, 2012.

[17] Z. Erkin, M. Beye, T. Veugeri, and R. Lagendijk, "Efficiently computing private recommendations," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2011.

[18] M. Barni, P. Failla, R. Lazzeretti, A. Sadeghi, and T. Schneider, "Privacy-preserving ecg classification with branching programs and neural networks," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 2, pp. 452 –468, june 2011.

[19] M. Roughan and J. Arnold, "Data fusion without data fusion: localization and tracking without sharing sensitive information," in *Inform., Dec. Con., 2007. IDC'07*, 2007, pp. 136–141.

[20] M. Roughan and J. Arnold, "Multiple target localisation in sensor networks with location privacy," in *Sec. and Priv. in Ad-hoc and Sen. Net.*, pp. 116–128. Springer, 2007.

[21] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Adv. Crypt. EUROCRYPT '99*, 1999, pp. 223–238.

[22] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proc. 41st ACM symp. theory comp.*, 2009, pp. 169–178.

[23] A. Yao, "Protocols for secure computations," in *IEEE Symp. on Found. of Comp. Science (FOCS'82)*, 1982.

[24] A. Yao, "How to generate and exchange secrets," in *IEEE Symp. on Found. of Comp. Science (FOCS'86)*, 1986.

[25] R. Lazzeretti and M. Barni, "Private computing with garbled circuits [applications corner]," *IEEE Signal Process. Mag.*, vol. 30, no. 2, pp. 123–127, 2013.

[26] V. Kolesnikov, A. Sadeghi, and T. Schneider, "A systematic approach to practically efficient general two-party secure function evaluation protocols and their modular design," *J. Comp. Sec.*, vol. 21, no. 2, pp. 283–315, 2013.

[27] S. Even, O. Goldreich, and A. Lempel, "A randomized protocol for signing contracts," *Comm. ACM*, vol. 28, no. 6, pp. 647, 1985.

[28] D. Beaver, "Precomputing oblivious transfer," in *Adv. Crypt. – CRYPTO'95*. 1995, vol. 963 of *LNCS*, pp. 97–109, Springer.

[29] R. Lazzeretti and M. Barni, "Division between encrypted integers by means of garbled circuits," in *Int. Workshop on Inf. Forensics Security (WIFS)*. IEEE, 2011, pp. 1–6.

[30] T. Veugen, "Encrypted integer division," in *Int. Workshop on Inf. Forensics Security (WIFS)*. IEEE, 2010, pp. 1–6.