Evaluating Encryption Schemes for Use in Secure Multi-Party VoIP Conferencing

Introduction

Secure, real-time, n-party VoIP teleconferencing is an area of encrypted communication which the market has largely failed to actualize for private or commercial use. In this analysis, we discuss different encryption regimes that could be utilized by a product developed for secure voice communication. We discuss the benefits, costs, studies, and working implementations of Homomorphic Encryption (HE), as well as different varieties of Secure Multi-Party Computation (MPC) in the context of a secure teleconference bridge. At the time of writing this piece, no cited implementation has surpassed the early stages of testing and development. However, we claim that if a product for secure, multiparty teleconferencing between 2-50 parties were to be developed, it should utilize the linear-shared computation variant of MPC for the best results of both security and functionality.

Overview: HE vs MPC

Homomorphic Encryption (HE) is a cryptographic regime that preserves relationships between plaintext inputs before, during, and after ciphertext transformation [1]. Practically speaking, the decrypted result of any computation done on homomorphically-encrypted data will exactly be the unencrypted inputs had they undergone the same computation. This characteristic of HE allows the data to undergo a variety of different operations without the data ever being decrypted, which permits calculations to be done by a third party without ever revealing the original input data. There are several ways to implement homomorphic encryption; each implementation carries its own unique properties. For example, Fully Homomorphic Encryption (FHE) supports any number of mathematical operations on encrypted data [1], but can be computationally expensive [2]. Likewise, Partially Homomorphic Encryption (PHE) allows for only addition or multiplication (but not both) to be performed on the encrypted data [1], at a lesser computational overhead [3]. In this analysis, we will mostly be discussing a variant of FHE that primarily deals with the addition operation, which we will refer to as Additive Homomorphic Encryption (AHE). The allowed mathematical operations and computational costs of AHE are feasible for secure VoIP conferencing [4].

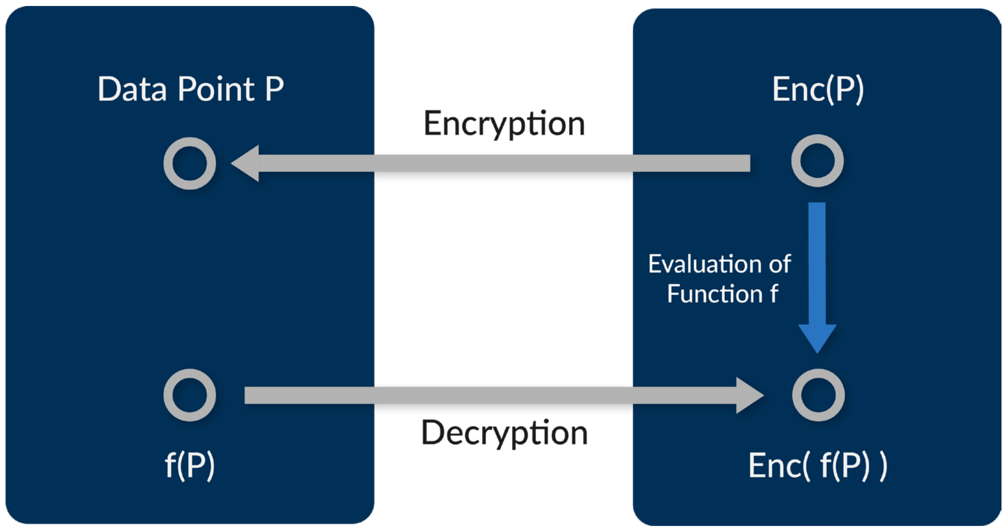

Figure 1. Mathematical operations performed on homomorphically-encrypted data, when decrypted, will exactly resemble the original input had it undergone the same operation.

Secure Multi-Party Computation (MPC) is defined as the act of computing a function from inputs given by multiple parties and revealing the output of the function to all participating clients while keeping individual inputs private [5]. For example, if two parties wanted to compare their individual sums of money without disclosing the amount of each (i.e. Yao's Millionaire Problem), secure MPC could provide the means of doing so, eliminating the need for a trusted third party [6]. Like HE, there are several different ways to effectively perform MPC, but we will be discussing the following two implementations: Garbled Circuits (GC) and Linear Secret Sharing (LSS). Garbled circuits are convoluted Boolean circuits, where it is difficult to observe what values are being passed gate-to-gate. The circuits are built in such a way that they effectively "dissolve" after their use, requiring a new circuit to be built for each calculation [7]. Linear Secret Sharing enables clients to break their inputs into several pieces or shares. Each share of a party's input does not contain enough information to reveal the entire secret input. The input shares are then sent to a collection of servers or other computing bodies, where shares from all parties are used in a collaborative computation. After the computation is complete, all parties will receive an output from the computing body, with the secrecy of each party's input still conserved [8].

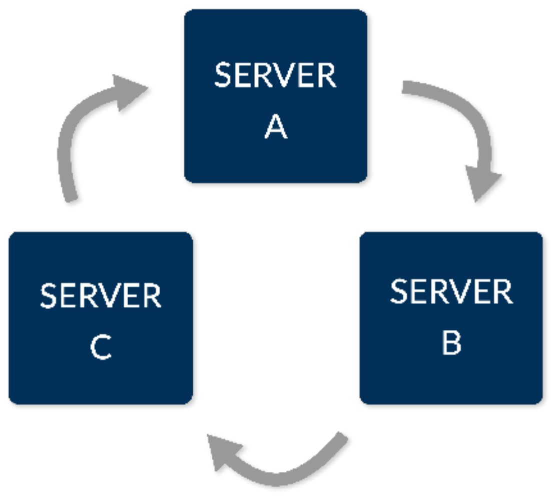

Figure 2. Input shares from each client are sent to a collection of servers, which jointly evaluate the resulting output. The links between servers, as discussed in [8], are protected by SSL or some other method.

HE & MPC State of the Art: Benefits and Issues in Secure Voice

Although procedurally different, all these methods can be used to achieve the same goal of secure computation with multiple parties. GC is a perfectly feasible option for Secure MPC. However, most recent innovations have been pertaining to two-party systems and not n-party systems as would be required for secure VoIP [7]. We will discuss the future of GC in a later section, but will temporarily table further analysis of it moving forward.
A variant of FHE that reduces most or all operations to addition, also referred to as AHE, can theoretically be used to achieve secure, multi-party voice conferencing and has been demonstrated in [4] to be executable on a relatively small scale. In practical use, several different clients connect to a VoIP server from a variety of access points. Each client encrypts their audio and sends it to the server. The server performs homomorphic mixing operations on encrypted audio received from all participating clients and then sends the result back to each party, where it is decrypted and played back. Input data from one client is not excluded in the output said client receives, so an extra mechanism is needed to avoid echoing. This end-to-end encryption ensures that only the participating parties (i.e., the clients with valid keys) can actively participate in the conference.
The benefit of using a server instead of point-to-point encryption manifests in the scalability. Without use of a server, the number of encryption links needed grows quadratically, severely degrading call quality even with few participating parties [4]. In the case of the implementation used by [4], it is claimed that clients could generate 16-bit audio streams at 48 kHz and that call quality was "substantially better than PSTN." Likewise in [4], 80 bits of security were achieved after setting the system parameters. Special conditions for the parameters of the system (plaintext size, ciphertext size, VoIP sample rate, etc.) needed to be set so end-to-end lag of the system was kept under the classical limit of 150 ms.

Although a successful implementation of secure VoIP by way of this additive HE scheme (AHE) has been shown to be physically feasible, it carries a wealth of issues in both security and performance. For security, it is likely that an AHE VoIP scheme will not see vast increases past the previously mentioned 80-bit security anytime in the upcoming future. This is due to the shallow depth requirement of the circuits needed for successful secure conferencing [4]. To increase the bits of security, a deeper circuit would have to be utilized, which, in this case, requires use of a multiplication function. Current multiplication functions in lattice-based HE schemes do not run fast enough to be considered viable for secure VoIP. If a method were to be developed that increased multiplication runtime by one or two orders of magnitude [4], deeper circuits would be physically viable, which would provide increases to overall security and protection against particular attack methods like noise injection from an ill-intentioned client, which is hypothesized to be largely effective against this scheme in its current state. To construct even deeper circuits to perform operations like voice recognition, bootstrapping evaluation would have to be implemented. Bootstrapping is a technique used to analyze ciphertexts with noise values too large to decrypt normally. The runtime of bootstrapping in HE is magnitudes slower than even the multiplication function, ultimately making circuits of this depth unattainable, barring drastic new implementations or schemes of HE [2].

On the performance of AHE VoIP, it is worth noting that the produced implementation saw feasible results for small parties; in particular, the maximum sample size which was used was a party of 7 clients [4]. Although no serious voice distortions were reported with 7 participating clients, it was seen experimentally that active talkers needed to speak "more slowly than normal" in order to avoid distortion by latencies [4]. This system was also optimized for a maximum of four active speakers at a time, not because of physical limitation, but for the ability of all participating clients to keep track of the conversation. In the maturity survey conducted by [7], current estimates indicate that secure computations with variants of FHE are most appropriate in the use of academic prototypes, and not the open market.

Linear Secret Sharing (LSS) is the act of representing a secret input as shared values, where no single shared value contains enough information to reveal the content of the entire input [8]. Input shares are encrypted, sent to a computing body (multiple servers, in this case), and added to component shares contributed by other parties. Each server computes its own share of the function output and communicates its value to a neighboring server. Pre-computations of lookup tables are distributed across each server at random indices, greatly reducing the time needed for each computation while still preserving the privacy of each input [8]. After a computation cycle is complete, the body of servers sends the result to each participating client, where the result will be decrypted and played back [8]. For secure voice conferencing, these LSS calculations require a minimum of three servers in order to function optimally [8]. The advantage of LSS is not only being able to jointly compute a function based on private inputs, but doing so while no single computing body (server) contains enough information to reveal individual inputs or the entirety of the conference if it becomes compromised. In the implementation described in [8], each server repeated the joint computation n times, where n is the number of participating clients. The reason for this is to negate the need for echo cancellation; each client had their own individual input removed before receiving output from the servers. As described in [8], each client generates a pair of keys for each server being used in the computation: one for audio streams sent to the server and another for streams received by the server. The security benefits of LSS come from the distribution of data over multiple servers and the fact that data are encrypted with respect to multiple keys from multiple clients. There are several implementations of LSS which can improve security and increase its capability beyond the "honest-but-curious" model employed by [8].

The implementation of Smart, Pastro, Damgard, and Zakarias or SPDZ regime, as detailed by [9], is computationally similar to the working implementation described by [8] and provides the ability to easily detect any dishonest participating parties. Likewise, the SPDZ regime utilizes pre-computations, which would scale the online calculation to any number of participating clients. Conferencing via LSS is not without its issues, however. Although no implementation has applied filtering to defend against injected noise from a nefarious client, it may be possible to implement filtering without the computational overhead that the HE scheme requires [9]. There are two working implementations described by the work of [8]. The first was an audio teleconference performed with four clients. Although the call was of high-quality, there were unwanted artifacts present, such as audible clicks and other transient sounds. The second experiment conducted was streaming recorded music from one client to a second client, with an audience of approximately 60 listening to the resulting output [8]. The quality of the playback was likened to a radio broadcast, with distortions coming provably from spikes in network latency. Scalability is a concern for both LSS and HE. Both of our state-of-the-art examples do not account for any testing on the scale of clients we are looking for. However, literature surrounding LSS suggests that it can be done feasibly [7], [8], [9]. In the maturity survey conducted by [7], it was determined that both active-security and passive-security LSS applications and products are ready for the open market.

Comment on Costs: Computation and Added Security

Computational overhead is a concern for both HE and LSS. For the HE scheme, there is a very clear relationship: deeper circuit = longer computation time. As stated before, order to achieve more security or features like voice recognition, homomorphic multiplication must be applied, which would increase the computation cost to the point of being unusable for VoIP [4]. As detailed in [8] and [9], there are several pre-computations that can take place to reduce the amount of time spent on the online phase of computation. Several potential methods of this pre-computing exist, so further research in this specific area would prove worthwhile. For security, LSS has several points of discussion. Multiple client keys and server distribution immediately come to mind for honest-but-curious defense; this leaves LSS still potentially vulnerable to cheating, such as a party inputting an algorithm or query instead of voice audio. In [9], there is a detailed description of covert security, which would make known any dishonest party attempting to cheat the system. Although not directly identical, the secure SPDZ scheme and the system described in [8] are computationally and functionally similar, making integration of parts from one into the other quite possible [8]. But the fact that both active- and passive-security LSS schemes are estimated to be ready for the open market is a large advantage over HE, which was deemed useful only for academic purposes in its current state [7].

HE & MPC: The Future of Each

The discussed encryption schemes have vast room for further innovation and development, but improvements in one may be easier to achieve than improvements in the other. As discussed in [4], there are several issues with performing this kind of task in the HE scheme, but one of the biggest issues that arises is real-time multiplication of encrypted data. Evaluating multiplication functions is extremely costly in this kind of HE regime and, with such large computational overhead, would make any audio communicating infeasible. To achieve more bits of security and thwart attacks like noise injection, homomorphic multiplication would need to see a speedup by about one or two orders of magnitude [4]. To see features like voice recognition, homomorphic multiplication would need to see speedup on the order of four or five magnitudes [4]. Barring an extreme, new HE regime or a new implementation of homomorphic multiplication, this is unlikely to change in the near future. As long as this multiplication stagnation is present in the realm of HE VoIP, the secure, large-scale conferencing we are after does not appear possible.

More possibilities seem to exist when analyzing the LSS regime. LSS has already been implemented in the famous case of the 2008 Danish Sugar Beet Auction, and has seen large improvements since then [10]. As discussed in [8], several improvements are still being discovered which can potentially provide up to an order of magnitude of better performance. For our purposes, most of the innovation will come in the form of reducing overall computation time and having the ability to include n participating parties. From [8] and [9], it appears very possible to achieve the innovations we are looking for by finding ways to integrate pre-multiplication functions, but will require further research and trials to be sure. GC has somewhat recently been shown to be a viable option for n-party computations. But for n>2, this technology is still in its infancy, especially for secure voice [11]. GC may one day be considered a viable option for the task at hand, but as of the time of writing, there is too little practical evidence for it to be considered for feasible secure VoIP.

Post-Quantum Computing

Both encryption schemes analyzed depend on key sharing at some point in their performance. Their resistivity against the looming threat of quantum computing will depend on what sort of scheme is used for the key generation. As detailed in [12], there are certain regimes which will no longer be secure in the era of quantum computing (RSA, DSA, Elliptic Curve), and there are certain regimes which will still be considered safe as long as longer key sizes are used (AES and SHA). These post-quantum estimations should be considered when encrypting audio data and when generating keys for decryption. In the implementation discussed in [4], 80 bits of AES security were achieved. By current forecasts, quantum computing is expected to render AES-128 obsolete in the near future, suggesting that shallow-circuit HE should not be pursued as a long-term solution for secure VoIP [13]. The amount of security provided by MPC is dependent on how one sets the parameters of their system. The LSS scheme we have discussed utilizes public key algorithms like RSA, which, at a sufficient length (longer than 2048-bit) are expected to resist quantum attacks in the immediate and intermediate future [13].

Conclusions

Based on the available evidence and an analysis of the performance of current encryption schemes for market-ready, secure, scalable VoIP conferencing, we determine that MPC by way of LSS is the most effective scheme to implement. Although a working prototype has been developed using end-to-end HE, there are large concerns with security, scalability, and overall performance. Despite LSS having similar concerns with its implementations, there are numerous areas for improvement, whereas HE seems to be stuck on multiplication and not viable for real-time voice communicating. As of writing this piece, LSS gives the best chance of achieving our multi-party VoIP conferencing goal, complete with the potential for scalability and adequate security. Although more research and experimentation is needed, secure, scalable conferencing is possible through LSS. Development of a product of this nature could be used in any situation where secure communication is necessary. From conversations between governments, to tactical movement on the battlefield, to exchanging sensitive information with key personnel, a product made for secure, scalable conferencing would not only make these tasks easier, but also safer and more efficient.

References

[1] Mohan, Maya, et al. "Homomorphic Encryption-State of the Art." 2017 International Conference on Intelligent Computing and Control (I2C2), 2017, doi:10.1109/i2c2.2017.8321774.

[2] Gentry, Craig, et al. "Homomorphic Evaluation of the AES Circuit." Lecture Notes in Computer Science Advances in Cryptology – CRYPTO 2012, 3 Jan. 2015, pp. 850–867., doi:10.1007/978-3-642-32009-5_49.

[3] Ogburn, Monique, et al. "Homomorphic Encryption." Procedia Computer Science, vol. 20, 2013, pp. 502–509. Elsevier, doi:10.1016/j.procs.2013.09.310.

[4] Rohloff, Kurt, et al. "Scalable, Practical VoIP Teleconferencing With End-to-End Homomorphic Encryption." IEEE Transactions on Information Forensics and Security, vol. 12, no. 5, 2017, pp. 1031–1041., doi:10.1109/tifs.2016.2639340.

[5] Orlandi, Claudio. "Is Multiparty Computation Any Good in Practice?" 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2011, doi:10.1109/icassp.2011.5947691.

[6] Du, Wenliang, and Mikhail J. Atallah. "Secure Multi-Party Computation Problems and Their Applications." Proceedings of the 2001 Workshop on New Security Paradigms - NSPW 01, Sept. 2001, doi:10.1145/508171.508174.

[7] Archer, David W., et. al. "Maturity and Performance of Programmable Secure Computation." IEEE Security & Privacy, vol. 14, no. 5, 27 Oct. 2016, pp. 48-56., doi:10.1109/msp.2016.97.

[8] Launchbury, John, et al. "Application-Scale Secure Multiparty Computation." Programming Languages and Systems Lecture Notes in Computer Science, 2014, pp. 8–26., doi:10.1007/978-3-642-54833-8_2.

[9] Damgård, Ivan, et al. "Practical Covertly Secure MPC for Dishonest Majority – Or: Breaking the SPDZ Limits." Lecture Notes in Computer Science Computer Security – ESORICS 2013, 2013, pp. 1–18., doi:10.1007/978-3-642-40203-6_1.

[10] Bogetoft, Peter, et al. "Secure Multiparty Computation Goes Live." Financial Cryptography and Data Security Lecture Notes in Computer Science, 2009, pp. 325–343., doi:10.1007/978-3-642-03549-4_20.

[11] Ben-David, Assaf, et al. "FairplayMP – A System for Secure Multi-Party Computation." Proceedings of the 15th ACM Conference on Computer and Communications Security - CCS 08, 2008, doi:10.1145/1455770.1455804.

[12] Chen, Lily, et al. "Report on Post-Quantum Cryptography." NIST, Apr. 2016, doi:10.6028/nist.ir.8105.

[13] "Commercial National Security Algorithm Suite and Quantum Computing FAQ." Cryptome.org, NSA, CSS, and IAD, Jan. 2016, cryptome.org/2016/01/CNSA-Suite-and-Quantum-Computing-FAQ.pdf