Poster: MPClan: Protocol Suite for Privacy-Conscious Computations

Nishat Koti*, Shravani Patil*, Arpita Patra*, Ajith Suresh†
*Indian Institute of Science, Bangalore, Email: {kotis, shravanip, arpita}@iisc.ac.in
†Technical University of Darmstadt, Email: suresh@encrypto.cs.tu-darmstadt.de

***Abstract*—The growing volumes of data being collected and its analysis to provide better services are creating worries about digital privacy. To address privacy concerns and give practical solutions, the literature has relied on secure multiparty computation. However, recent research has mostly focused on the small-party honest-majority setting of up to four parties, noting efficiency concerns. In this work, we extend the strategies to support a larger number of participants in the honest-majority setting with efficiency at the center stage. Cast in the preprocessing paradigm, our semi-honest protocol improves the online complexity of the decade-old state-of-the-art protocol of Damgård and Nielson (CRYPTO'07). In addition to having an improved online communication cost, we can shut down almost half of the parties in the online phase, thereby saving up to 50% in the system's operational costs. Our maliciously secure protocol also enjoys similar benefits and requires only half of the parties, except for one-time verification, towards the end. To showcase the practicality of the designed protocols, we benchmark popular applications such as deep neural networks, graph neural networks, genome sequence matching, and biometric matching using prototype implementations. Our improved protocols aid in bringing up to 60-80% savings in monetary cost over prior work.**

Our contributions We improve the practical efficiency of n-party honest-majority protocols using *function-dependent* preprocessing [1], [2], [3], [4], [5], [6]. Our protocol suite, MPClan, follows a 3-tier architecture (Fig. 1) to attain the goal of privacy-conscious computations.

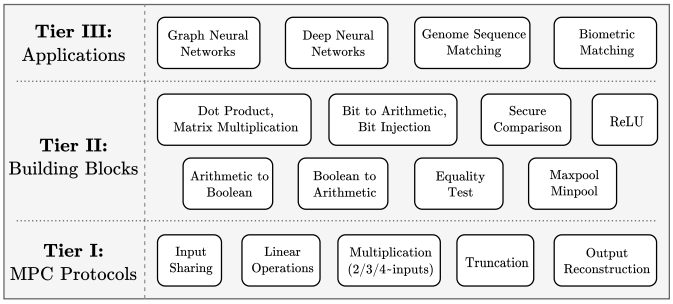


Fig. 1: Hierarchy of primitives in our 3-tier framework

MPC protocols Our goal is to design protocols with a fast online phase. Thus, working over $\mathbb{Z}_{2^\ell}$ and relying on RSS, we design a semi-honest MPC protocol assuming a one-time shared-key setup for correlated randomness. Our protocol requires communicating only $2t$ ring elements in the online phase and t in the preprocessing for a multiplication gate. We are the first to achieve a communication cost of $2t$ in the online phase (unlike $3t$ in the prior works [7], [8]), without incurring any overhead in the total cost, i.e., our total cost still matches that of the best known (optimized) semi-honest honest-majority protocol [7], [8]. We extend our protocol to provide malicious security with *fairness*[1] at the cost of additionally communicating t elements in the online phase and $2t$ in the preprocessing phase. Although (*abort*[2]) protocol of [9] has the same communication as our maliciously secure protocol, we achieve a stronger security notion of fairness. Moreover, our protocol avoids the consistency check after each level of circuit evaluation, reducing the number of rounds by $\mathcal{O}(d)$ (d denotes circuit depth).

We benchmark our semi-honest and malicious protocols over synthetic circuits comprising one million multiplications with varying depths of 1, 100, and 1000, where gates are distributed equally across each level in the circuit. We compare against optimized ring variant of DN07 [10]. The online phase of our semi-honest protocol enjoys the benefits of pushing 33% communication to a preprocessing phase compared to DN07, which corroborates improvement in our protocol's online complexity. Our malicious protocol retains the online communication cost of DN07 while incurring a similar overhead in the preprocessing. With respect to online run-time, our semi-honest protocol's time is expected to be similar to DN07.

Compared to the semi-honest protocol, the malicious variant incurs a minimal overhead of less than one second in online run-time due to a one-time verification phase. However, the overhead is higher (10 seconds) for the case of the overall run-time due to the distributed zero-knowledge proof computation in the preprocessing phase and gets amortized for deeper circuits. Another key highlight of our protocols is their improved monetary cost, as evident from Fig. 3. Concretely, for nine parties (semi-honest), we observe a saving of 17% over DN07 for a depth-1 circuit, and it increases up to 72% for circuits with depth 1000. This is primarily due to the reduction in the number of online parties

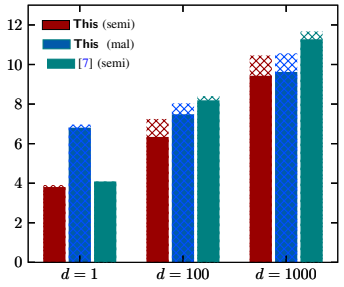
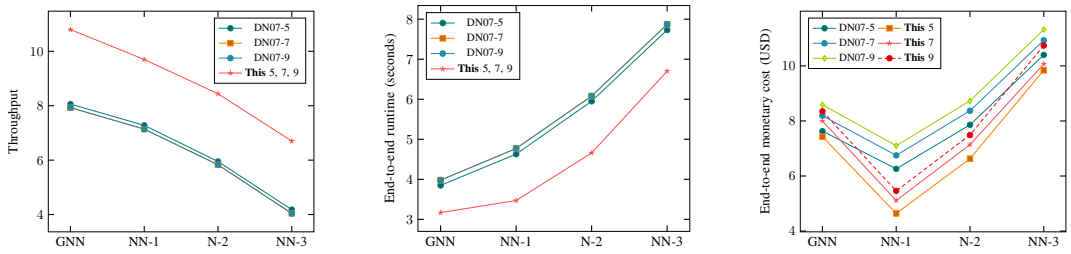


Fig. 3: Monetary cost (in USD for 1000 instances for $n = 9$ parties) for circuit evaluation of various depths (d), reported in $\log_2$ scale. Solid bars - computation over network with asymmetric round trip time (rtt), crosshatch - additional cost incurred with symmetric rtt.

[1]Guarantees either all parties receive the output or none do.
[2]Honest parties may not receive the output while corrupt parties do.



(a) Online throughput (b) End-to-end runtime (c) End-to-end monetary cost

Fig. 2: Comparison for GNN and deep NN between our semi-honest protocol and DN07 (values plotted are logarithmic in base 2)

over DN07. Comparing our semi-honest and malicious variants, the latter has an overhead of $8\times$ for depth-1 circuit, and it reduces to $1.14\times$ for depth-1000 circuit due to the amortization kicking in for deeper circuits. Interestingly, our malicious variant outperforms even the semi-honest DN07 upon reaching circuit depths of 100 and above. As for throughput, our semi-honest variant witnesses up to $1.78\times$ improvements in TP (for a single execution) over DN07 in the asymmetric rtt, which vanishes in the symmetric rtt setting. While moving from semi-honest to malicious security, we observe a significant drop in TP, which is about $3\times$ for the depth-1 circuit. This is due to the increased run time owing to the verification in the online phase for the malicious setting. However, this drop tends to zero for deeper circuits (as verification cost gets amortized), making the online phase of our malicious protocol on par with that of semi-honest.

Applications To showcase the practicality of our framework and improvements of our protocols, we benchmark the following applications in the WAN setting using Google Cloud instances. Owing to the inherent restrictions of RSS and keeping the focus on practical scenarios, we showcase the performance of our protocols for $n = 5, 7$, and 9 and compare with the state-of-the-art (optimized) protocol of DN07 [7].

1. *Graph neural network (GNN).* Inference phase of graph neural network [11], [12] is benchmarked on MNIST [13] data set. We see an improvement of around $7\times$ in online run-time and up to $180\times$ in online communication. Up to 15% savings are observed in monetary cost compared to DN07.

2. *Deep neural networks (NN).* Inference phases of deep neural networks such as LeNet [14] and VGG16 [15] are benchmarked. While monetary cost savings are up to 71%, up to $6\times$ improvement in online run-time and throughput are observed. Semi-honest results for GNN, NN appear in Fig. 2.

3. *Genome sequence matching.* Our similar sequence queries (SSQ) protocol for secure genome matching is based on the edit distance approximation protocol of [16], [17]. In comparison to [7], we witness improvements of up to $5\times$ in online run-time and throughput, as reported in Table I when the number of sequences in database (m) is 2000 and block length (ω) is 30 [16]. For the monetary cost, our semi-honest protocol saves up to 65% over DN07, and malicious has 42% overhead over semi-honest counterpart.

Ref.	n	Online			End-to-end		
		Comm[a]	Time	TP[b]	Comm[c]	Time[d]	Cost[e]
DN07	5	407.23	86.29	44.50	0.40	86.29	0.33
	7	610.85	92.97	41.30	0.60	92.97	0.46
	9	814.46	92.99	41.01	0.80	92.99	0.60
This (semi)	5	15.39	17.61 ±.02	217.93 ±0.2	0.40	21.69 ±.02	0.11
	7	23.08			0.60		0.16
	9	30.78			0.80		0.21
This (mal)	5	22.79	18.3 ±.2	209.84	0.42	34.52 ±.2	0.17
	7	33.88		209.49	0.64		0.25
	9	44.06		207.23	0.85		0.30

[a]communication in MB [b]TP denotes throughput [c]communication in GB [d]Time in seconds [e]monetary cost in USD

TABLE I: Genome sequence matching for $m = 2000, \omega = 30$.

REFERENCES

[1] H. Chaudhari, A. Choudhury, A. Patra, and A. Suresh, "ASTRA: High Throughput 3PC over Rings with Application to Secure Prediction," in *ACM CCSW@CCS*, 2019.

[2] M. Byali, H. Chaudhari, A. Patra, and A. Suresh, "FLASH: Fast and Robust Framework for Privacy-preserving Machine Learning," *PETS*, 2020.

[3] A. Patra and A. Suresh, "BLAZE: Blazing Fast Privacy-Preserving Machine Learning," in *NDSS*, 2020.

[4] H. Chaudhari, R. Rachuri, and A. Suresh, "Trident: Efficient 4PC Framework for Privacy Preserving Machine Learning," in *NDSS*, 2020.

[5] N. Koti, M. Pancholi, A. Patra, and A. Suresh, "SWIFT: Super-fast and Robust Privacy-Preserving Machine Learning," in *USENIX Security*, 2021.

[6] N. Koti, A. Patra, R. Rachuri, and A. Suresh, "Tetrad: Actively Secure 4PC for Secure Training and Inference," To Appear in NDSS, 2022.

[7] I. Damgård and J. B. Nielsen, "Scalable and unconditionally secure multiparty computation," in *CRYPTO*, 2007.

[8] D. Genkin, Y. Ishai, M. M. Prabhakaran, A. Sahai, and E. Tromer, "Circuits resilient to additive attacks with applications to secure computation," in *STOC*, 2014.

[9] D. Escudero and A. Dalskov, "Honest Majority MPC with Abort with Minimal Online Communication," in *LATINCRYPT*, 2021.

[10] D. Boneh, E. Boyle, H. Corrigan-Gibbs, N. Gilboa, and Y. Ishai, "Zero-Knowledge Proofs on Secret-Shared Data via Fully Linear PCPs," in *CRYPTO*, 2019.

[11] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering," in *NeurIPS*, 2016.

[12] L. Shen, X. Chen, J. Shi, Y. Dong, and B. Fang, "An Efficient 3-Party Framework for Privacy-Preserving Neural Network Inference," in *ESORICS*, 2020.

[13] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010, http://yann.lecun.com/exdb/mnist/.

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. of the IEEE*, 1998.

[15] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *ICLR*, 2015.

[16] T. Schneider and O. Tkachenko, "EPISODE: efficient privacy-preserving similar sequence queries on outsourced genomic databases," in *AsiaCCS*, 2019.

[17] G. Asharov, S. Halevi, Y. Lindell, and T. Rabin, "Privacy-preserving search of similar patients in genomic data," *PETS*, 2018.

Poster: MPClan: Protocol Suite for Privacy-Conscious Computations

Nishat Koti*, Shravani Patil*, Arpita Patra*, and Ajith Suresh**
*Indian Institute of Science Bangalore, India
**TU Darmstadt, Germany

Secure Multi Party Computation (MPC)

- Introduced by Andrew Chi Chi Yao [1982]
- Enables **n** mutually distrusting parties to jointly compute a public function on their private inputs.
- Properties
  - Privacy: Nothing beyond function output is leaked
  - Correctness: All parties obtain the correct output of the function
- Adversarial model (Models the distrust among the parties)
  - Semi-honest: honest but curious
  - Malicious: arbitrarily deviate from protocol specification
- Corruption threshold
  - Honest Majority: majority are honest
  - Dishonest Majority: minority are honest
- Security levels
  - Security with Abort: honest parties may abort without receiving output
  - Fairness: either all parties or none get the output
  - Guaranteed Output Delivery (GOD): all parties guaranteed to obtain output

Highlights

- Model: n parties, t corrupt (t<n/2), semi-honest and malicious security
- Multiplication (semi-honest): online - **2t** ring elements, preprocessing - **t** ring elements
- 33% improvement in online communication over state-of-the-art optimized ring protocol of DN07 [1, 2]
- Multiplication (malicious): online - **3t** ring elements, preprocessing - **3t** ring elements

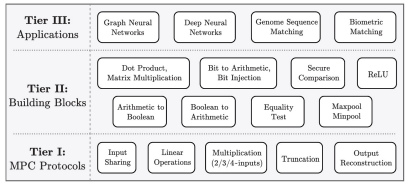
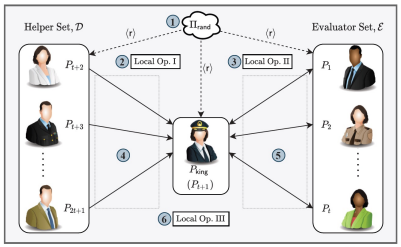


Figure 1: Hierarchy of primitives in our 3-tier framework

MPClan Protocol

- Facilitates secure outsourced and non-outsourced computation setting
- Works over l-bit rings (64-bit for benchmarks) in preprocessing paradigm
- Semi-honest multiplication:



① Generation of random $r \in \mathbb{Z}_{2^\ell}$ ② Computing $[r]$ & $\langle r \rangle$ ③ Computing $\mathcal{E}[\lambda_a], \mathcal{E}[\lambda_b]$ ④ $\mathcal{D}$ sending $\{[z-r]\}_{\mathcal{D}}$ to P_{king} ⑤ $\mathcal{E}$ sending $\{[z-r]\}_{\mathcal{E}}$ to P_{king} and receiving result from P_{king} ⑥ Computing $\langle\langle z \rangle\rangle$

Figure 2: Steps of multiplication protocol

Benchmarking

- Using ENCRYPTO library in C++17
- Google Cloud (n1-standard-64 instances)
- Runtime, Communication, Throughput
- Monetary cost (using Google Cloud pricing)
- #parties: 5, 7, and 9

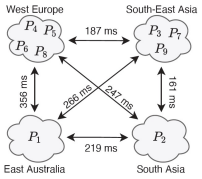


Figure 3: Round trip time (rtt)

Comparison with DN07 [1, 2]

- Monetary cost: **72%** saving
- Throughput: **1.78x** gain

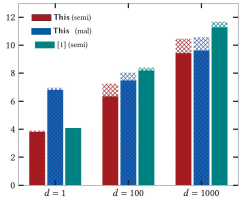


Figure 4: Monetary cost (in USD) for evaluating circuits (1000 instances) of various depths (d) for $n = 9$ parties. The values are reported in $\log_2$ scale. Bars in solid colors denote computation over network given in Fig. 3, while the area represented via crosshatch pattern denotes the additional cost incurred in the symmetric rtt setting (356 ms).

Applications

- Neural networks (NN) - Inference
- Graph neural network (GNN) Inference
- Similar sequence queries (SSQ) for genome sequence matching

NN and GNN architectures:

- NN-1: 3-layered fully connected network with ReLU activation after each layer (around 118K parameters)
- NN-2: LeNet[3] has 2 convolutional layers and 2 fully connected layers with ReLU activation after each layer, additionally followed by maxpool for convolutional layers (around 431K parameters)
- NN-3: VGG16[4] has $16 layers in total and comprises of fully-connected, convolutional, ReLU activation and maxpool layers (around 138 million parameters)
- GNN[5,6]: Graph convolution layer followed by ReLU activation and a fully connected layer with 10 nodes

Ref.	n	Online			End-to-end		
		Comm[a]	Time	TP[b]	Comm[c]	Time[d]	Cost[e]
DN07	5	407.23	86.29	44.50	0.40	86.29	0.33
	7	610.85	92.97	41.30	0.60	92.97	0.46
	9	814.46	92.99	41.01	0.80	92.99	0.60
This (semi)	5	15.39	17.61 ±.02	217.93 ±0.2	0.40	21.69 ±.02	0.11
	7	23.08			0.60		0.16
	9	30.78			0.80		0.21
This (mal)	5	22.79	18.3 ±.2	209.84 209.49 207.23	0.42	34.52 ±.2	0.17
	7	33.88			0.64		0.25
	9	44.06			0.85		0.30

[a]communication in MB [b]TP denotes throughput [c]communication in GB [d]Time in seconds [e]monetary cost in USD

Table 1: Genome sequence matching for $m = 2000$, $\omega = 30$.

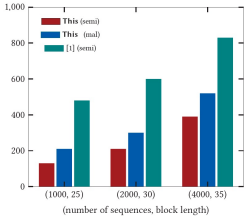
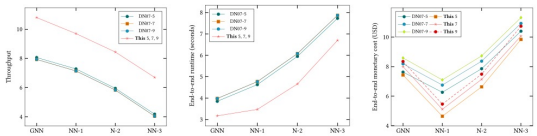


Figure 5: Monetary cost for SSQ evaluation for varying number of sequences and block lengths ((1000,25), (2000, 30), (4000,35)) for $n = 9$ parties. Costs for 1000 instances are reported in USD.



(a) Online throughput (b) End-to-end runtime (c) End-to-end monetary cost (1000 queries)

Figure 6: Comparison for GNN and deep NN between our semi-honest protocol and DN07 (values plotted are logarithmic in base 2)

Contact

Nishat Koti
Indian Institute of Science, Bangalore
Email : kotis@iisc.ac.in
Phone : +91 9049018189

References

1. Ivan Damgård, and Jesper Buus Nielsen. Scalable and unconditionally secure multiparty computation. In CRYPTO 2007.
2. Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Zero-Knowledge Proofs on Secret-Shared Data via Fully Linear PCPs. In CRYPTO 2019.
3. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient based learning applied to document recognition. Proc. IEEE (1998), 2278–2324.
4. K. Simonyan, and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014).
5. Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In NeurIPS 2016.
6. Liyan Shen, Xiaojun Chen, Jinqiao Shi, Ye Dong, and Binxing Fang. An Efficient 3-Party Framework for Privacy-Preserving Neural Network Inference. In ESORICS 2021.