Secure Medical Image Analysis with CrypTFlow

Javier Alvarez-Valle MSR UK
Pratik Bhatu MSR India
Nishanth Chandran MSR India
Divya Gupta MSR India
Aditya Nori MSR UK

Aseem Rastogi MSR India
Mayank Rathee MSR India
Rahul Sharma MSR India
Shubham Ugare UIUC, USA

Abstract

We present CRYPTFLOW, a system that converts TensorFlow inference code into Secure Multi-party Computation (MPC) protocols at the push of a button. To do this, we build two components. Our first component is an end-to-end compiler from TensorFlow to a variety of MPC protocols. The second component is an improved semi-honest 3-party protocol that provides significant speedups for inference. We empirically demonstrate the power of our system by showing the secure inference of real-world neural networks such as DENSENET121 for detection of lung diseases from chest X-ray images and 3D-UNet for segmentation in radiotherapy planning using CT images. In particular, this paper provides the first evaluation of secure *segmentation* of 3D images, a task that requires much more powerful models than classification and is the largest secure inference task run till date.

1 Introduction

Secure multiparty computation (MPC) allows a set of mutually distrusting parties to compute a publicly known function on their secret inputs without revealing their inputs to each other. This is done through the execution of a cryptographic protocol which guarantees that the protocol participants learn only the function output on their secret inputs and nothing else. MPC has made rapid strides - from being a theoretical concept three decades ago [13, 32], to now being on the threshold of having real world impact. One of the most compelling use cases for MPC is machine learning (ML) - e.g. being able to do secure ML inference when the model and the query are private inputs belonging to different parties. There has been a flurry of recent works aimed at running inference securely with MPC such as SecureML [21], MinioNN [18], ABY^3 [20], CHET [11], SecureNN [30], Gazelle [16], Delphi [19], and so on. Unfortunately, these techniques are not easy-to-use by ML developers and have only been demonstrated on small deep neural networks (DNNs) on tiny datasets such as MNIST and CIFAR. However, in order for MPC to be truly ubiquitous for secure inference tasks, it must be both easy to use by developers with no background in cryptography and capable of scaling to the DNNs used in practice.

In this work, we present CRYPTFLOW, a system that converts TensorFlow [4] inference code into MPC protocols at the push of a button. By converting code in standard TensorFlow, a ubiquitous ML framework that is used in production by various technology companies, to MPC protocols, CRYPTFLOW significantly lower the entry barrier for ML practitioners and programmers to use cryptographic MPC protocols in real world applications. We make the following contributions:

First, for the developer frontend, we provide a compiler, called *Athos*, from TensorFlow to a variety of secure computation protocols (both 2 and 3 party) while preserving accuracy. The compiler is designed to be modular and it provides facilities for plugging in different MPC protocols. To demonstrate this modularity, we have integrated Athos with the following backends: ABY-based [12] 2-party

PPML NeurIPS 2020 Workshop, Vancouver, Canada.

computation (2PC), SCI-based 2PC [25], Aramis-based malicious secure 3-party computation [17], and Porthos-based semi-honest secure 3-party computation (3PC).

Second, for the cryptographic backend, we provide a semi-honest secure 3-party computation protocol, *Porthos*, that outperforms all prior protocols for secure inference and enables us to execute, for the first time, cryptographically secure inference of ImageNet scale networks. Prior work in the area of secure inference has been limited to small networks over tiny datasets such as MNIST or CIFAR. We have evaluated CRYPTFLOW on secure inference over DNNs that are at least an order of magnitude larger than the state-of-the-art [5, 7, 8, 11, 16, 18–21, 26, 27, 30]. Even on MNIST/CIFAR, Porthos has lower communication complexity and is more efficient than prior works [20, 26, 30].

Third, we demonstrate the ease-of-use, efficiency and scalability of CRYPTFLOW by evaluating on RESNET50 [14] for ImageNet classification, DENSENET121 [15] for detection of lung diseases from chest X-ray images and 3D-UNet [28] for segmentation of raw 3D CT images.

Our toolchain is publicly available[1]. This paper reviews the original CRYPTFLOW paper [17] briefly and its increment lies in the secure segmentation evaluation (Section 5.3).

2 Athos

Athos compiles TensorFlow inference code to secure computation protocols. The transformations implemented in Athos are sensitive to the performance of MPC protocols. For performance reasons all efficient secure computation protocols perform computation over fixed-point arithmetic - i.e., arithmetic over integers or arithmetic with fixed precision. Athos automatically converts TensorFlow code over floating-point values into code that computes the same function over fixed-point values. This compilation is done while *matching* the inference accuracy of floating-point code. Prior works ([16, 18–21, 30]) in the area of running ML securely have performed this task by hand with significant losses in accuracy over floating-point code.

Athos represents a 32-bit floating-point number r by a 64-bit integer $\lfloor r.2^s \rfloor$ for a precision or scale s. Then operations on 32-bit floating-point numbers are simulated by operations on 64-bit integers. For example $r_1 \times r2$ is simulated as $\frac{\lfloor r_1.2^s \rfloor \times \lfloor r_1.2^s \rfloor}{2^s}$. A large s causes integer overflows and a small s leads to accuracy loss. To obtain a suitable scale s (all variables have the same precision in Athos output), Athos works by "sweeping through" various precision levels to estimate the best precision [17].

3 Porthos

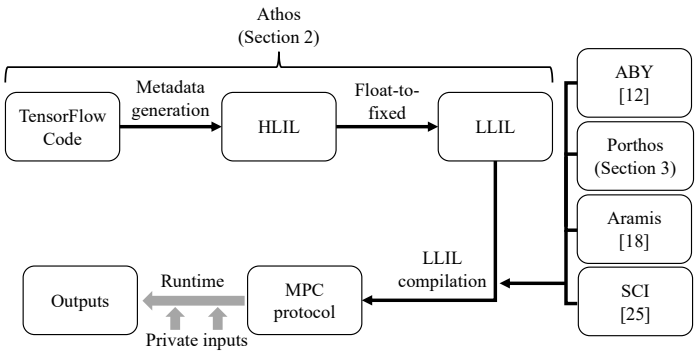
Porthos is an improved semi-honest 3-party secure computation protocol (tolerating one corruption) that builds upon SecureNN [30]. Porthos makes two crucial modifications to SecureNN. First, SecureNN reduces convolutions to matrix multiplications and invokes the Beaver triples [6] based matrix multiplication protocol. When performing a convolution with filter size $f \times f$ on a matrix of size $m \times m$, the communication is roughly $2q^2f^2 + 2f^2 + q^2$ elements in the ring $\mathbb{Z}_{2^{64}}$, where $q = m - f + 1$. Porthos computes these Beaver triples by appropriately reshaping $m \times m$ and $f \times f$ matrices. This reduces the communication to roughly $2m^2 + 2f^2 + q^2$ ring elements. Typically the filter size, f, is between 1 and 11 and the communication of Porthos can be up to two orders of magnitudes less than SecureNN. Additionally, in SecureNN, the protocols for non-linear layers (such as ReLU and MaxPool) require the third party to send secret shares to the first two parties. In Porthos, we cut this communication to half by eliminating the communication of one of these shares [17]. This reduces the communication in the overall ReLU and MaxPool protocols by 25%.

4 Motivating Example

In this section, we describe the end-to-end working of CRYPTFLOW through an example of logistic regression. The toolchain is shown in Figure 1.

CRYPTFLOW takes as input a pre-trained floating-point TensorFlow model. For example, consider the code snippet for logistic regression over MNIST dataset in TensorFlow as shown in Figure 2. Our compiler first generates the TensorFlow graph dump as well as metadata containing the dimensions

[1] `https://github.com/mpc-msri/EzPC`




```
# x is (1,784) MNIST image.
# W and b are model parameters.

print(tf.argmax(tf.matmul(x, W) + b, 1))
```

Figure 2: Logistic Regression: TensorFlow snippet

Figure 1: CRYPTFLOW: End-to-end toolchain

```
xW = MatMul(x, W);
xWb = MatAdd(xW, b);
output(ArgMax(xWb));
```

(a)

```
xW = MatMul(x, W);
ScaleDown(xW, 15); //15 bit precision
xWb = MatAdd(xW, b);
output(ArgMax(xWb));
```

(b)

Figure 3: Logistic Regression in (a) floating-point: HLIL syntax (b) fixed-point: LLIL syntax

of all the tensors. Next, the TensorFlow graph dump is compiled into a high-level intermediate language HLIL. The code snippet for logistic regression in HLIL is shown in Figure 3a. Next, Athos' float-to-fixed converter translates the floating-point HLIL code to fixed-point code in a low-level intermediate language LLIL. This step requires Athos to compute the right precision to be used for maximum accuracy Figure 3b shows the LLIL code snippet for logistic regression. The operation `ScaleDown(X, s)` divides each 64-bit integer entry of tensor X by 2^s. The function calls in the LLIL code can be implemented with a variety of secure computation backends - e.g. ABY [12] for the case of 2-party secure computation, Porthos for the case of semi-honest 3-party secure computation, and Aramis [17] for the malicious secure variant. Different backends provide different security guarantees and hence vary in their performance. For this example, the three backends take 227ms, 6.5ms, and 10.2ms respectively.

5 Experiments

Overview. First, in Section 5.1, we use CRYPTFLOW for secure classification on ImageNet using the following pre-trained TensorFlow models: RESNET50[2] and DENSENET121[3]. We show that the fixed-point MPC protocols generated by Athos matches the accuracy of cleartext floating-point RESNET50 and DENSENET121. We also show how the optimizations in Porthos help it outperform prior works in terms of communication complexity and overall execution time. Finally, we discuss two case-studies of running CRYPTFLOW on DNNs for medical image analysis. The compilation time of CRYPTFLOW is around 5 sec for RESNET50, 35 sec for DENSENET121 and 2 minutes for 3D UNet.

5.1 Secure Inference on ImageNet

These experiments are in a LAN setting on 3.7GHz machines, each with 4 cores and with 16 GB of RAM. The measured bandwidth between each of the machines was at most 377 MBps and the latency was sub-millisecond.

RESNET50 takes 25.9 seconds and 6.9 GB of communication; DENSENET121 takes 36 seconds and 10.5 GB of communication. We measure communication as total communication between all 3 parties

[2] `https://github.com/tensorflow/models/tree/master/official/r1/resnet`

[3] `https://github.com/pudae/tensorflow-densenet`

Benchmark	Float Top 1	Fixed Top 1	Float Top 5	Fixed Top 5
ResNet50	76.47	76.45	93.21	93.23
DenseNet121	74.25	74.33	91.88	91.90

Table 1: Accuracy of fixed- vs floating-point.

Benchmark	SecureNN (s)	Porthos (s)	SecureNN Comm. (GB)	Porthos Comm. (GB)
ResNet50	38.36	25.87	8.54	6.87
DenseNet121	53.99	36.00	13.53	10.54

Table 2: Porthos vs SecureNN.

- each party roughly communicates a third of this value. We show that Athos generated fixed-point code matches the accuracy of floating-code on ResNet50 and DenseNet121 in Table 1.

Detailed comparisons of CrypTFlow with prior works on secure inference can be found in [17]. However, since Porthos builds on SecureNN, we compare them on ImageNet scale benchmarks in Table 2. For this purpose, we add the code of SecureNN available at [3] as another backend to CrypTFlow. These results show that Porthos improves upon the communication of SecureNN by a factor of roughly 1.2X–1.5X and the runtime by a factor of roughly 1.4X–1.5X.

5.2 Lung diseases from 2D chest X-Ray images

In [33], the authors train a DenseNet121 to predict lung diseases from chest X-ray images. They use the publicly available NIH dataset of chest X-ray images and end up achieving an average AUROC score of 0.845 across 14 possible disease labels. These DNNs are available as pre-trained Keras models. We converted them into TensorFlow using [2] and compiled the automatically generated TensorFlow code with CrypTFlow. During secure inference, we observed no loss in classification accuracy and the latency is similar to the runtime of DenseNet121 for ImageNet.

5.3 Segmenting tumors and organs at risk from 3D CT images

Half a million cancer patients receive radiotherapy each year [24]. Personalized radiation treatments require segmenting tumors and organs at risk from 3D volumetric images. Currently, this segmentation is a manual process where an oncologist draws contours along regions of interest slice-by-slice across the whole volume. This process often takes several hours per image which ML provided automation [22, 31] can reduce to minutes. We consider a 3D-UNet model [28] that takes as input a raw 3D image obtained via Computed Tomography (CT) scans of the pelvic region and delineates tumor volumes and organs at risk. This model's accuracy is within the inter-observer variability seen among clinical experts [23] and requires 1.87 Teraflops per inference.

Since this model is implemented in PyTorch, we first export it to ONNX and then use CrypTFlow's ONNX frontend. For our secure inference setup, each party has 32 cores running at 2.4GHz, no GPUs, and 128GB RAM. The parties are connected on a network with ping latency 0.2s and 625MBps bandwidth. On this set up, secure inference incurs a latency of 1 hour and 57 minutes and 557GB of communication. The most expensive operators in this computation are 3D transposed convolutions (or deconvolutions) and, to the best of our knowledge, CrypTFlow is the only secure inference tool that supports these operations. In our experience, it takes a couple of days for a scan to reach the oncologist for review and hence this latency overhead can be acceptable.

6 Related work and conclusion

Other related systems for converting PyTorch/Tensorflow to MPC protocols [1, 9, 10, 29] only support 3PC. Whereas, CrypTFlow additionally supports 2PC backends. CrypTFlow provides the first implementation and evaluation of a system for secure segmentation. With CrypTFlow, data scientists, with no background in cryptography, can obtain secure inference implementations for their trained models at the push of a button.

References

[1] Crypten by Facebook, 2019. URL `https://github.com/facebookresearch/CrypTen`.

[2] Keras to TensorFlow. `https://github.com/amir-abdi/keras_to_tensorflow`, 2019.

[3] SecureNN: 3-Party Secure Computation for Neural Network Training. `https://github.com/snwagh/securenn-public`, 2019.

[4] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Gregory S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian J. Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Józefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Gordon Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul A. Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda B. Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *CoRR*, abs/1603.04467, 2016. URL `https://arxiv.org/abs/1603.04467`.

[5] Nitin Agrawal, Ali Shahin Shamsabadi, Matt J. Kusner, and Adrià Gascón. QUOTIENT: Two-Party Secure Neural Network Training and Prediction. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, pages 1231–1247, 2019.

[6] Donald Beaver. Efficient Multiparty Protocols Using Circuit Randomization. In *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, pages 420–432, 1991.

[7] Fabian Boemer, Yixing Lao, Rosario Cammarota, and Casimir Wierzynski. nGraph-HE: A Graph Compiler for Deep Learning on Homomorphically Encrypted Data. In *Proceedings of the 16th ACM International Conference on Computing Frontiers, CF 2019, Alghero, Italy, April 30 - May 2, 2019*, pages 3–13, 2019.

[8] Nishanth Chandran, Divya Gupta, Aseem Rastogi, Rahul Sharma, and Shardul Tripathi. EzPC: Programmable and Efficient Secure Two-Party Computation for Machine Learning. In *IEEE European Symposium on Security and Privacy, EuroS&P 2019, Stockholm, Sweden, June 17-19, 2019*, pages 496–511, 2019.

[9] Morten Dahl, Jason Mancuso, Yann Dupis, Ben Decoste, Morgan Giraud, Ian Livingstone, Justin Patriquin, and Gavin Uhma. Private Machine Learning in TensorFlow using Secure Computation. *CoRR*, abs/1810.08130, 2018. URL `http://arxiv.org/abs/1810.08130`.

[10] Anders P. K. Dalskov, Daniel Escudero, and Marcel Keller. Secure evaluation of quantized neural networks. *Proc. Priv. Enhancing Technol.*, 2020(4):355–375, 2020.

[11] Roshan Dathathri, Olli Saarikivi, Hao Chen, Kristin Lauter, Saeed Maleki, Madan Musuvathi, and Todd Mytkowicz. CHET: An Optimizing Compiler for Fully-Homomorphic Neural-Network Inferencing. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2019, Phoenix, AZ, USA, June 22-26, 2019*, pages 142–156, 2019.

[12] Daniel Demmler, Thomas Schneider, and Michael Zohner. ABY - A Framework for Efficient Mixed-Protocol Secure Two-Party Computation. In *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015*, 2015.

[13] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 218–229, 1987.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016.

[15] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2261–2269, 2017.

[16] Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. GAZELLE: A Low Latency Framework for Secure Neural Network Inference. In *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*, pages 1651–1669, 2018.

[17] Nishant Kumar, Mayank Rathee, Nishanth Chandran, Divya Gupta, Aseem Rastogi, and Rahul Sharma. Cryptflow: Secure tensorflow inference. In *2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020*, pages 336–353. IEEE, 2020.

[18] Jian Liu, Mika Juuti, Yao Lu, and N. Asokan. Oblivious Neural Network Predictions via MiniONN Transformations. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 619–631, 2017.

[19] Pratyush Mishra, Ryan Lehmkuhl, Akshayaram Srinivasan, Wenting Zheng, and Raluca Ada Popa. Delphi: A Cryptographic Inference Service for Neural Networks. In *29th USENIX Security Symposium, USENIX Security 20*, Boston, MA, 2020.

[20] Payman Mohassel and Peter Rindal. ABY^3: A Mixed Protocol Framework for Machine Learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 35–52, 2018.

[21] Payman Mohassel and Yupeng Zhang. SecureML: A System for Scalable Privacy-Preserving Machine Learning. In *2017 IEEE Symposium on Security and Privacy, S&P 2017, San Jose, CA, USA, May 22-26, 2017*, pages 19–38, 2017.

[22] Stanislav Nikolov, Sam Blackwell, Ruheena Mendes, Jeffrey De Fauw, Clemens Meyer, Cían Hughes, Harry Askham, Bernardino Romera-Paredes, Alan Karthikesalingam, Carlton Chu, Dawn Carnell, Cheng Boon, Derek D'Souza, Syed Ali Moinuddin, Kevin Sullivan, DeepMind Radiographer Consortium, Hugh Montgomery, Geraint Rees, Ricky Sharma, Mustafa Suleyman, Trevor Back, Joseph R. Ledsam, and Olaf Ronneberger. Deep learning to achieve clinically applicable segmentation of head and neck anatomy for radiotherapy. *CoRR*, abs/1809.04430, 2018. URL `http://arxiv.org/abs/1809.04430`.

[23] Ozan Oktay, Jay Nanavati, Anton Schwaighofer, David Carter, Melissa Bristow, Ryutaro Tanno, Rajesh Jena, Gill Barnett, David Noble, Yvonne Rimmer, Ben Glocker, Kenton O'Hara, Christopher Bishop, Javier Alvarez-Valle, and Aditya Nori. Evaluation of Deep Learning to Augment Image-Guided Radiotherapy for Head and Neck and Prostate Cancers. *JAMA Network Open*, 3(11):e2027426–e2027426, 11 2020. ISSN 2574-3805. doi: 10.1001/jamanetworkopen.2020.27426. URL `https://doi.org/10.1001/jamanetworkopen.2020.27426`.

[24] Hubert Y Pan, Bruce G Haffty, Benjamin P Falit, Thomas A Buchholz, Lynn D Wilson, Stephen M Hahn, and Benjamin D Smith. Supply and demand for radiation oncology in the united states: Updated projections for 2015 to 2025. *Int J Radiat Oncol Biol Phys.*, 96(3):493–500, 2016. doi: 10.1016/j.ijrobp.2016.02.064.

[25] Deevashwer Rathee, Mayank Rathee, Nishant Kumar, Nishanth Chandran, Divya Gupta, Aseem Rastogi, and Rahul Sharma. Cryptflow2: Practical secure 2-party inference. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, CCS 2020*. ACM, 2020.

[26] M. Sadegh Riazi, Christian Weinert, Oleksandr Tkachenko, Ebrahim M. Songhori, Thomas Schneider, and Farinaz Koushanfar. Chameleon: A Hybrid Secure Computation Framework for Machine Learning Applications. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security, AsiaCCS 2018, Incheon, Republic of Korea, June 04-08, 2018*, pages 707–721, 2018. doi: 10.1145/3196494.3196522.

[27] M. Sadegh Riazi, Mohammad Samragh, Hao Chen, Kim Laine, Kristin E. Lauter, and Farinaz Koushanfar. XONN: XNOR-based Oblivious Deep Neural Network Inference. In *28th USENIX Security Symposium, USENIX Security 2019, Santa Clara, CA, USA, August 14-16, 2019*, pages 1501–1518, 2019.

[28] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015. URL `http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a`. (available on arXiv:1505.04597 [cs.CV]).

[29] Theo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. A generic framework for privacy preserving deep learning. *CoRR*, abs/1811.04017, 2018. URL `http://arxiv.org/abs/1811.04017`.

[30] Sameer Wagh, Divya Gupta, and Nishanth Chandran. SecureNN: 3-Party Secure Computation for Neural Network Training. *PoPETs*, 2019(3):26–49, 2019.

[31] Shuai Wang, Dong Nie, Liangqiong Qu, Yeqin Shao, Jun Lian, Qian Wang, and Dinggang Shen. CT male pelvic organ segmentation via hybrid loss network with incomplete annotation. *IEEE Trans. Medical Imaging*, 39(6):2151–2162, 2020. doi: 10.1109/TMI.2020.2966389. URL `https://doi.org/10.1109/TMI.2020.2966389`.

[32] Andrew Chi-Chih Yao. How to Generate and Exchange Secrets (Extended Abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167. IEEE Computer Society, 1986. doi: 10.1109/SFCS.1986.25.

[33] Xiaoyong Zhu, George Iordanescu, Ilia Karmanov, and Mazen Zawaideh, Mar 2018. URL `https://blogs.technet.microsoft.com/machinelearning/2018/03/07/using-microsoft-ai-to-build-a-lung-disease-prediction-model-using-chest-x-ray-images/`.