Facial Recognition with Post-Quantum Cryptography

NAU Northern Arizona University
College of Engineering, Informatics, and Applied Sciences

Charlie Herman, Kelly Marriott, Steven Perry

Sponsor: Dr. Tuy Nguyen

Problem Statement

Standard cryptography algorithms are susceptible to attacks using Shor's Algorithm, a powerful method of factoring very large integers using quantum computers, which can break most standard encryption. Alternate (Post-Quantum) means of data encryption are needed before large-scale quantum computers capable of performing these attacks become widely available. This is particularly problematic for the use case of video teleconferencing, wherein confidential topics may be discussed for business, government, or personal reasons. Additionally, facial images contain especially sensitive biometric information. A fast, secure solution for a common platform is needed to protect this data.

Proposed Solution

Our prototype solution is a Windows-based application that recognizes and extracts faces from images, video, or webcam feed, and implements post-quantum encryption and decryption of the faces. Additionally, parallel computing can be utilized to accelerate the process; cryptography can be performed in 3 modes: sequentially via CPU, parallel via CPU, or parallel via GPU.

Requirements and Constraints

Requirements:

- An application for Microsoft Windows
- Cryptography not susceptible to breaking via Shor's Algorithm
- Recognize and extract faces from still images, video, and streams
- Both CPU and GPU computing options for cryptographic processes

Constraints:

- Use Crystals-Kyber Post-Quantum cryptography
- Program developed using C/C++
- Use the OpenCV library for facial recognition
- Use the CUDA toolkit for the GPU implementation

Facial Recognition

Using the OpenCV Library and Haar-Cascade object recognition, an input image is divided into rectangles. Each rectangle is classified as either possibly containing a relevant feature (positive) or *not* containing a relevant feature (negative). Each positive rectangle is then compared to its neighbors to determine presence or absence of a target object (a face).

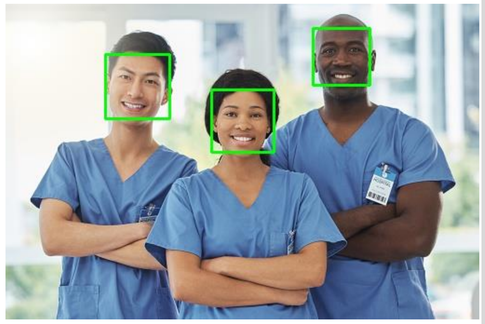
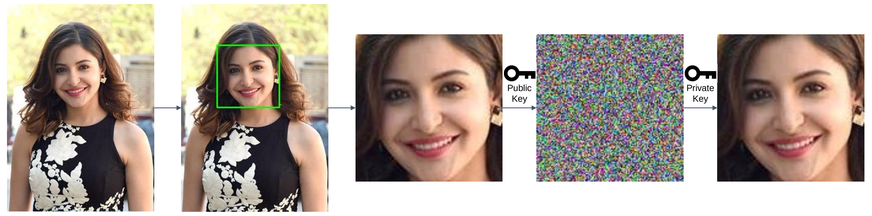


Image Analysis Example

```
Enter Kyber Implementation: 1. Kyber512 2. Kyber768     3. Kyber1024
        Enter:  1

Enter Processing Mode: 1. Single Threaded       2. Multi-Threaded (CPU) 3. Multi-Threaded (GPU)
        Enter:  2

Enter Input Mode: 1. Image      2. Video        3. Webcam
        Enter:  1

Enter input file, including extension (from Input folder):
        Enter:  102199.jpg

Enter Number of Threads to use (Even numbers recommended)
        Enter:  32

Time to encrypt file Output/102199.jpg_Image_Kyber512_Multi-Threaded (CPU) (32 threads)/Faces/face0.bmp (ms):    11
Time to decrypt file Output/102199.jpg_Image_Kyber512_Multi-Threaded (CPU) (32 threads)/Faces/face0.bmp (ms):     2
```

Console input is prompted to the user to select processing parameters such as which Kyber security mode to use, processing mode (single-threaded or multi-threaded), input mode (image, video, webcam), input file, and number of threads (if applicable). Encryption and decryption times are displayed as output in milliseconds. Results are stored in an output file.



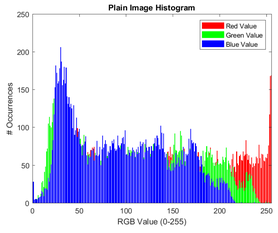
Step 1: User inputs an image via command line for processing.

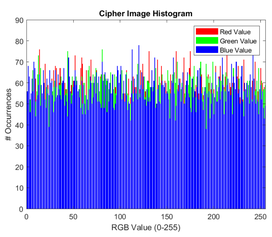
Step 2: OpenCV detects faces on the input image.

Step 3: Detected face is cropped and saved in output folder.

Step 4: Detected face is encrypted with Kyber, encrypted image is saved in output folder.

Step 5: Encrypted image is then decrypted with Kyber, decrypted image is saved in output folder.

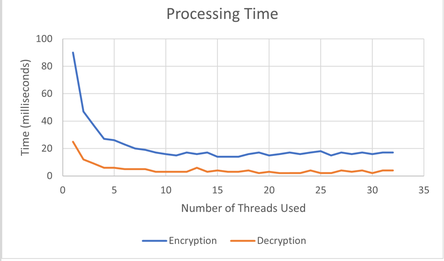


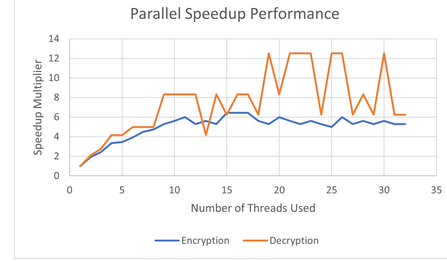


Step 6: Encrypted and decrypted images are analyzed and compared using MATLAB. Outputs include histograms and spreadsheet data for generated statistics.

Face #	Encryption Time (ms)	Decryption Time (ms)	entropy_red	entropy_green	entropy_blue	entropy_cipher_red	entropy_cipher_green	entropy_cipher_blue	CorrCoef_Red	CorrCoef_Green	CorrCoef_Blue	npcr_score	npcr_pVal	uaci_score	uaci_pVal
0	15	3	7.71	7.7569	7.5097	7.9866	7.9894	7.9885	0.0082528	-0.0080659	0.0081852	0.99597	0.34276	0.32651	9.28E-13

Results





These charts demonstrate the improvement in performance achieved by multi-threading (parallel processing). The input image used is the same as in the Image Analysis Example section, image "102199.jpg" from the NIST IJB-C dataset. The processing was performed using Kyber-512, and up to 32 threads were used.

Mean Encryption and Decryption Times (milliseconds)

Input file	# Faces Detected	512 encryption (1-thread)	512 encryption (32-thread)	512 decryption (1-thread)	512 decryption (32-thread)	768 encryption (1-thread)	768 encryption (32-thread)	768 decryption (1-thread)	768 decryption (32-thread)	1024 encryption (1-thread)	1024 encryption (32-thread)	1024 decryption (1-thread)	1024 decryption (32-thread)
450888.jpg	1	19.0	4.0	6.0	1.0	32.0	8.0	8.0	1.0	48.0	11.0	9.0	1.0
452226.jpg	2	26518.5	1363.5	6777.5	340.5	45465.5	2229.5	8957.5	1189.5	72210.5	3429.0	11291.5	1497.5
452414.jpg	3	30.3	3.0	8.3	0.7	50.7	5.0	10.3	0.7	76.0	76.0	13.7	13.7
455563.jpg	1	22.0	4.0	6.0	0.0	36.0	5.0	7.0	0.0	54.0	6.0	10.0	1.0
125.mp4	4362	519.4	71.0	118.8	16.0	870.1	118.8	157.4	20.9	1340.0	184.6	198.5	26.0
1362.mp4	78	59.8	4.8	15.4	1.2	107.9	7.8	21.8	1.6	185.4	12.1	29.7	1.9
6314.mp4	63	21.2	2.1	5.8	0.5	34.3	3.5	7.6	0.7	52.0	5.3	9.7	0.9
10626.mp4	23	38.6	3.7	10.5	1.0	62.6	69.0	14.0	1.2	95.8	9.3	17.2	1.3
16186.mp4	1605	283.5	38.6	64.6	8.8	473.2	64.4	85.4	11.5	729.4	99.8	107.4	14.3
32884.mp4	286	505.9	60.8	115.8	13.8	851.7	103.9	154.0	18.6	1324.9	164.6	195.2	23.5
35138.mp4	101	139.4	12.1	33.2	2.9	241.8	21.3	45.2	4.0	386.1	36.2	58.8	5.8
37248.mp4	242	525.1	55.3	121.0	12.6	889.7	94.9	161.8	17.1	1378.7	164.3	205.4	24.1
37904.mp4	1396	128.3	16.7	29.5	3.8	215.3	28.4	39.1	5.1	333.2	43.9	49.3	6.3
39642.mp4	2088	241.2	33.1	55.0	7.5	402.5	55.1	72.7	9.8	621.8	85.2	91.5	12.1
Combined Mean:		**787.8**	**92.5**	**369.8**	**23.5**	**2273.8**	**150.0**	**525.8**	**39.4**	**3751.9**	**229.1**	**676.2**	**56.3**
Parallel Improvement:		**88.3%**		**93.6%**		**93.4%**		**92.5%**		**93.9%**		**91.7%**	

**times displayed per face (ms)*

This table showcases the performance of the three security levels of Kyber as well as the speedup achieved through multi-threading. Times given are the average of all faces detected from a given input file (image and video).

Acknowledgements

Sponsor: Dr. Tuy Nguyen
Capstone Professor: Dr. Carlo da Cunha
Mentor: Jordan Beverly

References

[1] R. Avazi, J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schank, P. Schwabe, G. Seiler and D. Stehle, "CRYSTALS-Kyber Algorithm Specifications and Supporting Documentation, version 3.0," 2020.

[2] O. Regev, "On Lattices, Learning with Errors, Random Linear Codes, and Cryptography," School of Computer Science, Tel Aviv University, Tel Aviv, 2009.

[3] D. Stebila, "Introduction to post-quantum cryptography," in Summer School on Real-World Crypto and Privacy, Šibenik, Croatia, 2018.

[4] D. Balbas, "The Hardness of LWE and Ring-LWE: A Survey," IMDEA Software Institute, Madrid, 2021.