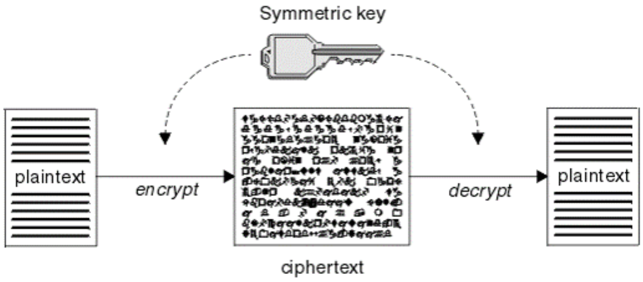
Krypton

Distributed Processing project

The project is to create a Python application that encrypts a long text using a symmetric key encryption algorithm RC4, utilizing multiprocessing to distribute the encryption task across *N* different worker machines connected through a network. The ciphertext will be saved to the file in the correct order and decrypted again by worker machines. The main functionality of the application can be depicted with the use of a simple scheme:



https://cryptobook.nakov.com/symmetric-key-ciphers

The design of the application will involve several distributed processing issues that will need to be addressed during the project implementation. One such issue is load balancing, as the encryption task will be distributed across multiple machines. Ensuring that each machine receives an equal share of the workload will be critical to maintaining high performance and minimizing bottlenecks.

Another important issue to consider is data synchronization. Since the encryption will be distributed across multiple machines, ensuring that the encrypted data is properly synchronized across all three machines will be necessary. This will require a robust communication protocol to enable the machines to coordinate their work and ensure the encrypted data is properly ordered.

Finally, the application will need to be designed with fault tolerance in mind. If one of the machines fails or goes offline during the encryption process, the application should be able to detect the failure and redistribute the workload to the remaining machines, without losing any data or compromising the overall security of the encryption.

To address these challenges, the application will use Python's multiprocessing library, which provides a convenient way to distribute tasks across multiple processes on different machines. Additionally, a robust communication protocol will be designed to ensure proper synchronization of encrypted data, and fault tolerance will be built at every level of the application to ensure that the encryption process can continue uninterrupted, even in the event of failure.