

**LAB 12**

**Part 2: Symmetric Key Cryptography**

**Copyright © (2022-2025) All Rights Reserved**

**Dr. Matthew Kisow**

VERSION 1

22-NOV-2024

**LAB**:NaCL Cryptography Programming – Part 2: Symmetric Key Cryptography

**OBJECTIVE**: Create a Python script that reads the symmetric key **part2.key.bin** and ciphertext **part2.ciphertext.bin** files, and performs decryption using the Salsa20 stream cipher to recover the original plaintext message.

**INSTRUCTIONS**: This script will decrypt the ciphertext in **part2.ciphertext.bin** using the symmetric key stored in **part2.key.bin**. The decryption is based on the Salsa20 stream cipher with Poly1305 message authentication code, ensuring both confidentiality and authenticity. Follow these steps to implement the **part2.py** script. Please follow the instructions for submitting this assignment on Blackboard.

1. **Prerequisites**
2. **Setting Up Your Python Environment**

Before you start programming your client, set up your Python environment:

1. Create a virtual environment:

Python3 -m venv .venv

1. Activate the virtual environment:

* **Linux/macOS:**

source .venv/bin/activate

* **Windows:**

.venv\Scripts\activate

1. Install the necessary dependencies:

pip install wheel pynacl requests

1. **Understand Salsa20 Decryption:**

* Encryption and decryption in Salsa20 use the same **SecretBox** method.
* The **decrypt** method of **SecretBox** verifies the integrity of the ciphertext before decryption, ensuring it hasn’t been tampered with.

1. **Writing the sym\_decrypter.py Script**
2. **Import Necessary Modules**

Use the **os** and **sys** , **pynacl** modules for file handling, and **SecretBox** from **nacl.secret** for decryption.

1. **Define Helper Functions**
   * **Read Binary Data from File:** Create a function to read binary data from the given file:

def read\_from\_file(filename):

"""Read binary data from a file."""

with open(filename, "rb") as file:

return file.read()

* + **Decrypt Using OTP:** Implement the XOR decryption logic:

def decrypt\_salsa20(ciphertext, key):

"""Decrypt ciphertext using the Salsa20 symmetric key."""

# Create a SecretBox with the given key

box = SecretBox(key)

# Decrypt the ciphertext and return plaintext

plaintext\_bytes = box.decrypt(ciphertext)

return plaintext\_bytes.decode('ascii')

1. **Build the Main Script Logic**
   1. Read the **part2.key.bin** and **part2.ciphertext.bin** files.
   2. Pass them to the decryption function.
   3. Print the recovered plaintext.
2. **Testing**
   1. Save the script as **part2.py**.
   2. Run it from the command line, providing the **part2.ciphertext.bin** and **part2.key.bin** files as arguments:

python3 part2.py part2.ciphertext.bin part2.key.bin

**DELIVERABLE**

Write a Python3 script named **part2.py** that communicates with the server, receives the ciphertext, and prints the plaintext message to the screen.