Assignment 09

cryptology – b keerthana

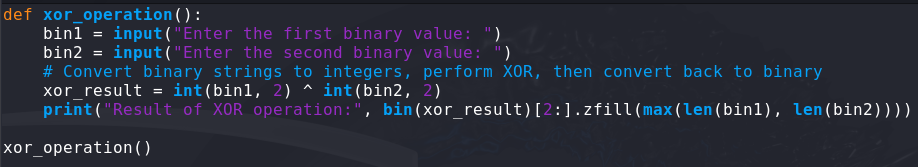
AMRUTHESH

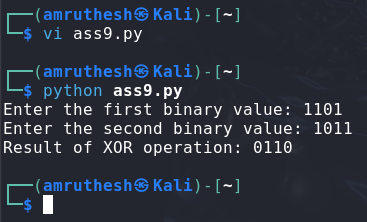
241059041

M.E – Cyber Security

MSIS, MANIPAL

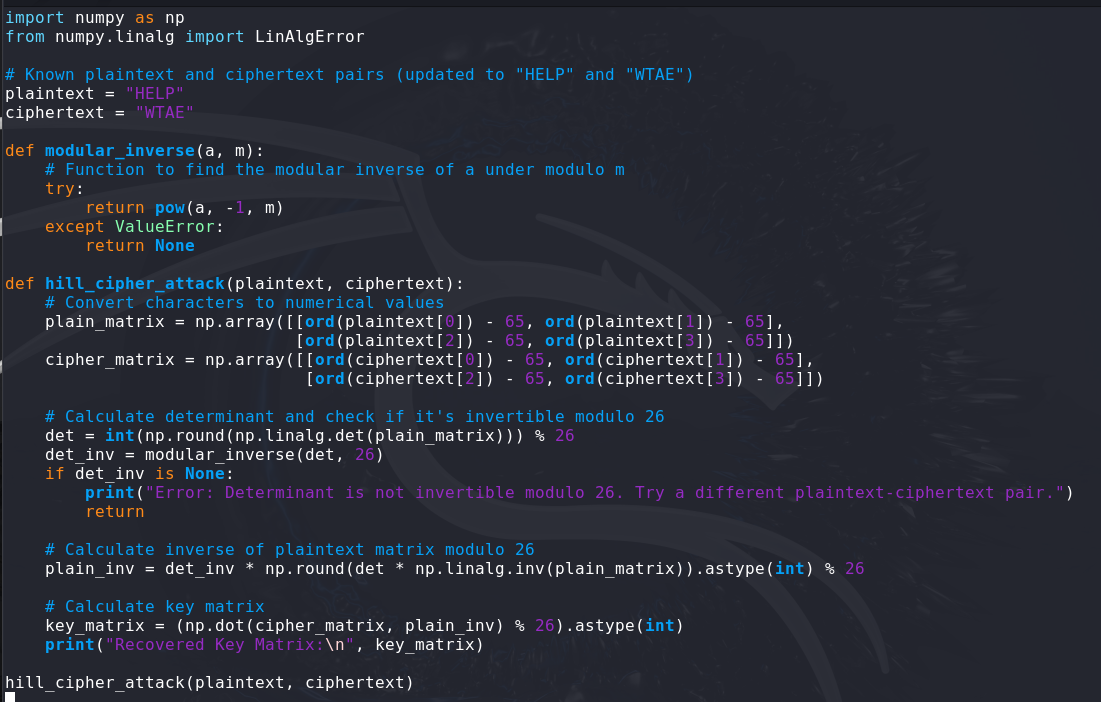
1. **Binary XOR Operation**: XOR (exclusive OR) is a bitwise operation used frequently in cryptographic applications, where each bit of the output is true only if one (but not both) of the bits being compared is true. This is often used for encrypting data as it can obscure data while allowing for easy reversibility when the key is known.

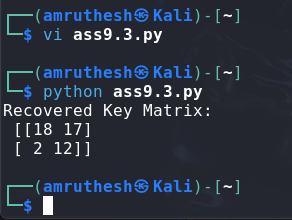




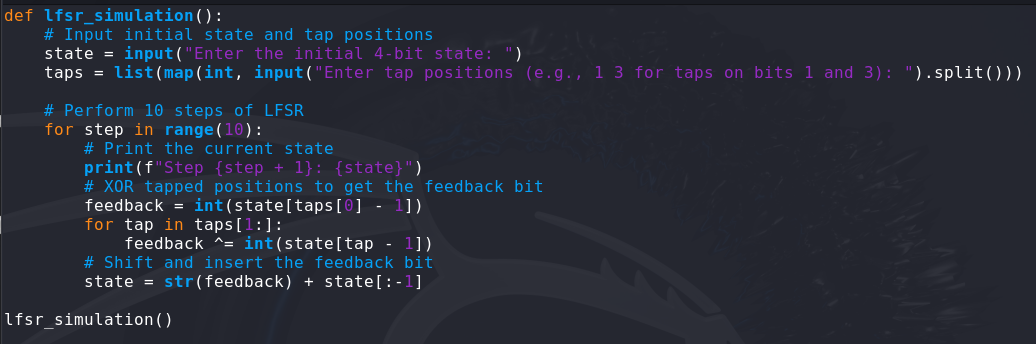
1. **Breaking Hill Cipher with Known-Plaintext Attack**: The Hill cipher, a classical polygraphic substitution cipher, can be broken using known plaintext and ciphertext pairs. In this task, given the pair MEET (plaintext) and URRG (ciphertext), the goal is to reverse-engineer the key matrix by solving a system of linear equations.

* The determinant of the matrix formed by the plaintext "MEET" was not invertible modulo 26, meaning it shared a common divisor with 26 (e.g., 13), making it impossible to calculate the inverse needed for decryption. In such cases, the Hill cipher decryption won't work unless the determinant is coprime with 26.





1. **4-bit LFSR (Linear Feedback Shift Register)**: LFSRs are used in stream ciphers and cryptographic algorithms to produce pseudo-random sequences. A simple LFSR generates a sequence based on a starting value (seed) and specific bit positions (taps) that feed back into the register, influencing future states.

****



1. **Report on Attacks on Linear Feedback Shift Registers (LFSR)**

**Introduction to LFSR**

Linear Feedback Shift Registers (LFSRs) are commonly used in cryptographic systems for generating pseudo-random sequences. An LFSR is a shift register that generates bits through a linear function of its previous states, often relying on XOR operations to produce a new bit based on a tap sequence. While efficient and simple, LFSRs are vulnerable to certain cryptographic attacks due to their linear properties.

**Types of Attacks on LFSR**

Several attacks target the LFSR’s weaknesses, with each exploiting the predictable patterns or limited randomness in the sequences generated. The main categories of attacks on LFSR-based systems include:

1. **Berlekamp-Massey Attack**: Uses the linear nature of LFSR to deduce the characteristic polynomial of the LFSR from observed outputs.
2. **Correlation Attack**: Exploits a statistical correlation between the output of a linear register and the original sequence to recover the LFSR states or keys.
3. **Algebraic Attack**: Uses algebraic equations derived from the LFSR’s outputs and taps to determine the key.
4. **Guess and Determine Attack**: Involves guessing part of the internal state and then using equations to determine the rest of the state.

**Detailed Explanation of Correlation Attack**

The **correlation attack** is a particularly effective attack on LFSR-based stream ciphers that is used when an LFSR output is combined with other LFSRs or non-linear components, resulting in a combined keystream. This attack exploits statistical correlations between the LFSR output and the final keystream to deduce parts of the initial state or key.

**How Correlation Attack Works**

1. **Correlation Detection**: The attack begins by analyzing the output keystream for correlations with the output of individual LFSRs. For example, if a keystream bit tends to equal an LFSR’s output bit with a probability slightly greater than 0.5, the attacker can detect this weak correlation.
2. **Hypothesis Testing**: Once a correlation is identified, the attacker can hypothesize the initial state of the LFSR by guessing the output bits. For each guess, they check if the generated keystream aligns statistically with the observed keystream.
3. **Verification**: If the guess produces a high correlation with the observed keystream, it is likely that the initial state is correct. If not, the attacker revisits their guess, adjusting bits until a correlation is found.

**Steps in a Correlation Attack**

* **Step 1**: Analyze the keystream for potential correlations. This may involve calculating the probability of each LFSR bit appearing in the final output keystream.
* **Step 2**: Formulate hypotheses about the LFSR’s initial state and simulate the LFSR’s output.
* **Step 3**: Compare the generated sequence with the observed keystream.
* **Step 4**: Narrow down the possibilities by focusing on hypotheses that show high correlation.
* **Step 5**: Once a hypothesis aligns well, the attacker can reconstruct the LFSR’s initial state or key.

**Countermeasures to Correlation Attack**

* **Use of Non-linear Combinations**: Combining multiple LFSRs using non-linear functions reduces the likelihood of detectable correlations.
* **Adding Noise**: Adding pseudo-random noise to the keystream can make it harder for attackers to detect correlations.
* **Use of Filters or Irregular Clocks**: Using filters or irregular clocking schemes for LFSRs introduces unpredictability, making it more challenging to establish a clear correlation.

**Conclusion**

The correlation attack demonstrates the vulnerability of LFSRs when used in a straightforward manner for cryptographic applications. To strengthen LFSR-based systems, cryptographers often combine LFSRs with non-linear functions, filters, or irregular clocking. Although LFSRs are efficient for generating pseudo-random sequences, these attacks reveal the importance of carefully designing cryptographic systems to prevent information leakage through predictable correlations.