

Assignment 1

February 11th, 2023

Programming Assignment¹

Pseudo-random Generator

[10]

Implement a provably-secure *pseudo-random generator*. Your goal is to create a pseudo-random generator which outputs a pseudo-random bit-string value of length $\ell(n)$ when the in-class function **generate()** is invoked.

Pseudo-random Function

[10]

Implement a provably-secure *pseudo-random function*. Create a keyed pseudo-random function that outputs a pseudo-random integer value when the in-class function **evaluate()** is invoked.

Encryption Scheme against Eavesdropping Adversary [5]

Implement a secure encryption-decryption scheme for an eavesdropping attack. You must implement both the encryption and the decryption functionalities as **two different** functions.

CPA-Secure Encryption Scheme

[15+10]

Implement a *variable-length CPA-secure* encryption-decryption scheme using block ciphers in **randomized counter mode** using the previously implemented PRF. Your goal is to output the *ciphertext* when the in-class function **enc()** is invoked and return the *plaintext* when the in-class function **dec()** is invoked. You can additionally attempt to implement other modes of operation, which will be considered as a part of **bonus**.

¹Only the constructions taught in class would be considered valid for the coding assignment.

Message Authentication Codes [20]

Implement a *variable-length* **message authentication code** scheme. You are supposed to return the *tag* when the in-class function **mac()** is invoked and return a boolean value when the function **vrify()** is invoked.

CBC-MAC [20]

Implement a *variable-length* **CBC-MAC** using the previously implemented PRF.²

CCA-Secure Encryption Scheme [20]

Implement a **CCA-secure** scheme using the **CPA** and **CBC-MAC** implementations. You are supposed to return the *cipher-text* when the in-class function **enc()** is invoked and return the *plain-text* (or not) when the in-class function **dec()** is invoked. Can we use the variable-length MAC construction as defined in Section 5? Justify.

Theory Assignment

Prove that the above constructions are *secure* in the *respective adversarial settings*.

Guidelines and Submission Format

Boilerplate

The programming assignment would be automatically graded, so it's crucial that you adhere to the **boilerplate conventions**.

Testing

The **input** and **output** files for your testing are provided here.

²Refer to pg. 121 in the textbook. It is required that you follow the third implementation.

Submission Format

Submit a zip file containing all the **codes** in respective directories with **PDFs** containing theoretical proofs to all the constructions. It must be named as *Rollno-A1.zip*. **The deadline for the final submission is 11:59 PM, 21st February.**

```
--init__.py
CBC_MAC
  CBC_MAC.py
  CBC_MAC.pdf
CCA
  CCA.py
  CCA.pdf
CPA
  CPA.py
  CPA.pdf
EAV
  EAV.py
  EAV.pdf
MAC
  MAC.py
  MAC.pdf
PRF
  PRF.py
  PRF.pdf
PRG
  PRG.py
  PRG.pdf
```