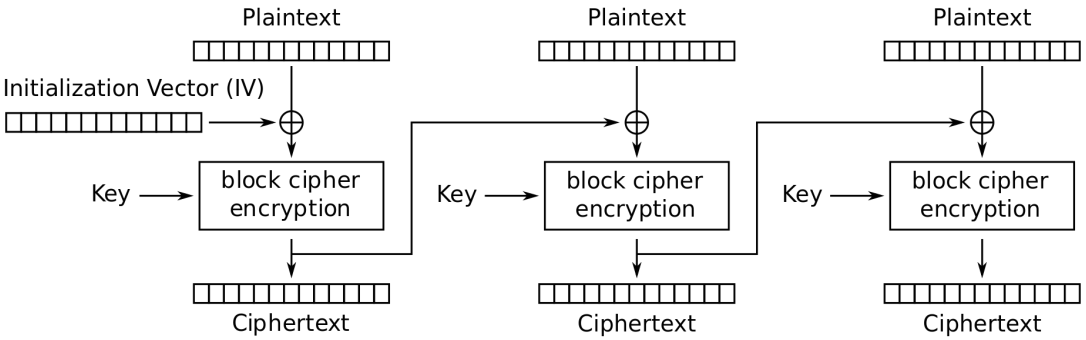
TEMA 1 - Securitatea Informatie

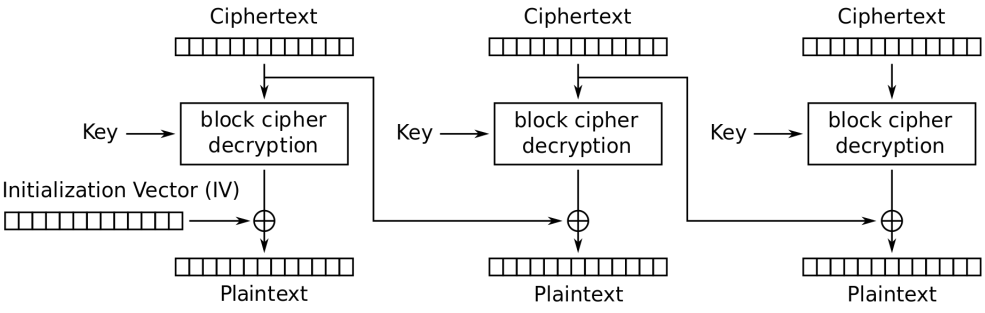
Asandoaiei David - IIIB4

November 2020

1 CBC criptari si decriptari



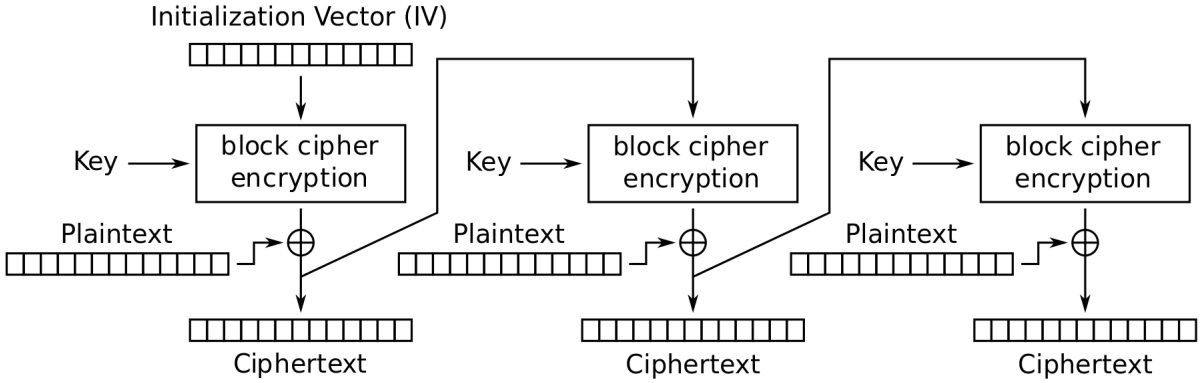
Cipher Block Chaining (CBC) mode encryption



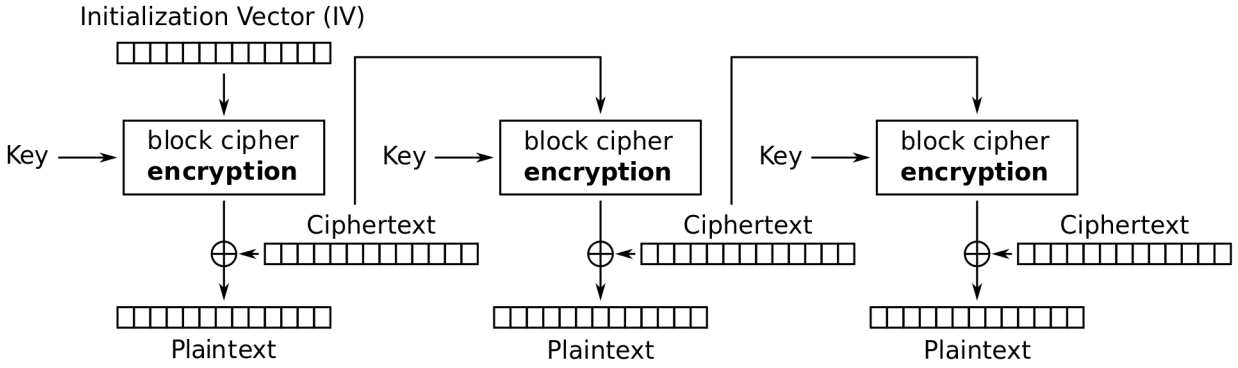
Cipher Block Chaining (CBC) mode decryption

ENCRYPTION:
Pentru criptarea CBC s-a padat plaintextul daca a fost nevoie pentru a ajugne la lungime divizibila cu 16bytes, dupa s-a impartit in blocuri de 16bytes, iar la fiecare pas s-a facut XOR intre blocul plain text si IV, rezultatul s-a criptat cu AES in modul ECB cu cheia respectiva, si astfel s a obtinut un bloc de ciphertext. La finalult fiecarei operatii de genul pe un bloc de date, IV ul ia valoarea blocului criptat de AES in modul ECB pentru a se face XOR ul dorit cu urmatorul bloc de date al Plaintextului.
La final se concateneaza blocurile de ciphertext si avem asadar mesajul nostru criptat in modul CBC.
DECRYPTION:
Pentru decriptarea CBC s-a impartit measjul criptat in blocuri de 16bytes, iar la fiecare pas s-a facut decriptarea blocului cu AES in modul ECB cu cheia respectiva, iar blocul rezultatul s-a supus operatiei de XOR cu IV obtinand astfel un bloc de plaintext. La finalul fiecarei astfel de iteratii, IV ul devine bucata de bloc de ciphertext cu care s-a lucrat la aceasta iteratie.La final se concateneaza toate blocurile de plaintext si se obtine mesajul decriptat, si daca este cazul se face unpadding.

2 CFB criptari si decriptari



Cipher Feedback (CFB) mode encryption



Cipher Feedback (CFB) mode decryption

ENCRYPTION:
Pentru criptarea CFB, se padeaza daca este cazul plaintextul pentru a ajunge la o lungime div cu 16bytes, dupa se imparte in blocuri de 16bytes.La fiecare pas, se realizeaza mai intai criptarea IV ului cu AES in modul ECB cu cheia respectiva si se obtine un nou bloc, iar cu acest bloc rezultat se face XOR pe bucata de plain text, iar blocul rezultat se adauga la ciphertext. La finalul fiecarei iteratii, IV ul devine blocul de ciphertext rezultat pentru calculele in urm iteratii. La finalul iteratiilor, mesajul criptat devine concatenarea tuturor blocurilor de ciphertext.
DECRYPTION:
Pentru decriptarea CFB, se imparte mesajul criptat in blocuri de 16 bytes, iar la fiecare iteratie se decripteaza IV ul folosind AES in modul ECB cu cheia corespunzatoare, iar cu blocul rezultat se face XOR cu bucata de ciphertext, rezultand o bucata de bloc plaintext. La finalul unei iteratii IV devine bucata de ciphertext cu care s-a lucrat la iteratia respectiva. La final mesajul decriptat este concatenarea bucatilor de plaintext, iar daca este cazul se face unpadding la acest mesaj.

3 Setup

Programul a fost scris in Python3, cu ajutorul environmentului PyCharm.
Ca librarie folosita in plus, s-a folosit " pycryptodome ", iar aceasta trebuie sa fie instalata cu ajutorul comenzii "pip3 install pycryptodome".

4 Rulare

Pentru a se rula programul se urmeaza urmatori pasi in ordine!
1. "python KeyManager.py" (se deschide serverul principal)
2. "python A.py" (se conecteaza nodul A la server)
3. "python B.py" (se conecteaza nodul B la server)

5 Program Files

Avem fisierul "KeyManager.py", care este serverul.
Avem fisierele "A.py" si "B.py" care sunt cei doi clienti.
Avem fisierul "cripto.py" unde se intampla toata magia criptarilor si celorlalte functii. Avem functii de criptare decriptare pt fiecare CBC resp CFB, avem functia de padding care padeaza daca este cazul cu "/0", avem functia de unpadding care daca s-a padat la inceput vom unpada la final, avem functia de criptare a cheii si iv ului pt a le transmite KM lui A si B,si de asemenea decriptarea acestora, avem functia de XOR care face xor intre toti bytesii trimisi ca arg, functia de output care concateneaza rezultatul alcatuit din blocurile de 16bytes.
Avem fisierul "additional.py" in care sunt specificate adresele si porturile pt realizarea conexiunilor ce vor avea loc, o functie ce realizeaza conexiunea la un server cu argumentele adresa si portul ce treb specificate, o functie de a deschide un fisier si a citi din el pt ca este fol ulterior in A, o functie de a scrie rezultatul decriptarii in fisierul "dec_secret.txt", o functie de a compara fisierul initial cu cel decriptat final.
Avem fisierul "secret.txt" unde se introduce ce se vrea trimis.
Avem fisierul "dec_secret.txt" unde este introdus textul decriptat dupa criptare pentru a vizualiza si valida rezultatele obtinute.
Conexiunea KM-A-B este facuta prin thread uri.

6 Functionalitatea Programului

1 Se porneste serverul KM (python KeyManager.py)
2 Se porneste client A (python A.py)
3 Se porneste client B (python B.py)
— adresa de conectare este "127.0.0.1", iar portul este 1234 (folositi pt server

si clientii A si B)—
4 A ii transmite lui KM modul de (CBC/CFB) dorit.
—A trebuie sa astepte sa se conecteze si B altfel nu va functiona transmiterea modului si va primi mesaj de la server cu "Te rog asteapta conectarea clientului B"
—de asemenea, A trebuie sa scrie modul de criptare dorit CBC/CFB exact "CBC" sau "CFB"; orice altceva scris, va primi un mesaj de eroare din partea server KM si va fi rugat sa scrie modul de criptare dorit CORECT!—
5 KM le transmite lui A si B doua mesaje identice cu cheia K respectiva si IV ul, acestea fiind criptate cu cheia K3, pe care toti 3 o au, acestia urmand sa decripteze.
6 A si B raspund cu un mesaj de confirmare criptat in modul ales. ("Sunt {A/B} si confirm primirea cheii si a vectorului de init)
7 KM decripteaza mesajele si le transmite un mesaj de inceput al comunicatiei catre cele doua noduri A si B
8 Nod B deschide un server pt A
—B isi deschide un server la adresa "127.0.0.1", dar la un port diferit de KM si anume 1235—
9 A se conecteaza la B
—A se conecteaza folosind adresa "127.0.0.1" si port specific 1235—
10 A se foloseste de fisierul "secret.txt" pentru a cripta
11 A ii trimite lui B fisierul criptat corespunzator iar lui KM nr de blocuri
12 B decripteaza ce a primit de la A , afiseaza mesajul decriptat si ii trimite lui KM numarul de blocuri criptate
13 KM compara numarul de blocuri de la A si B si afiseaza daca sunt egale
14 KM afiseaza daca fisierul initial coincide cu cel decriptat final
15 Program complet functional!

References

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation
SI labs from Google Classroom