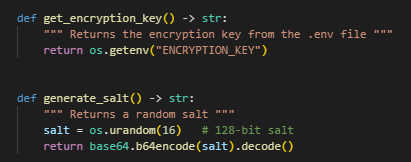
**بخش اول و دوم**

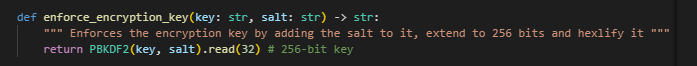
کلید (که در مرحله هشتم گفته شد برابر *AUT\*ICTSec\*2022* باشد) در فایل .env قرار داده شده و با استفاده از دو کتابخانه os و dotenv خوانده میشود. برای خواندن پسورد و ساختن یک salt رندوم دو تابع زیر تعریف شده اند:



همانطور که مشخص است، salt ساخته شده 128 بیت داشته و پسورد به صورت پیش فرض از فایل .env خوانده میشود.

**بخش سوم**

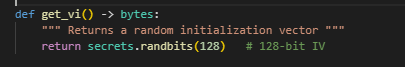
با استفاده از کتابخانه PBKDF2، دو رشته salt و key با یکدیگر ترکیب شده و یک رشته 256 بیتی خروجی داده میشود. این کد در قالب تابع زیر تعریف شده:



برای نمایش خروجی این تابع فایل main.py را اجرا کنید.

**بخش چهارم**

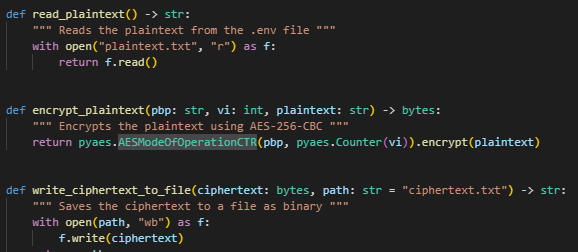
تابع get\_vi برای تولید مقدار بردار اولیه نوشته شده:



این تابع مقدار VI برابر 16 بایت (128 بیت) باز میگرداند که از جنس bytes است.

**بخش پنجم**

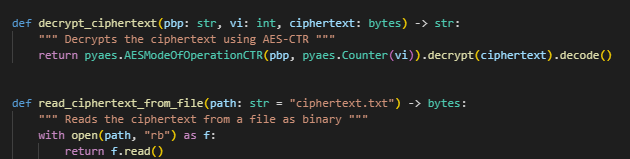
سه تابع برای رمزنگاری کردن رشته با استفاده از کلید و VI، تابع دیگر برای ذخیره سازی بایت باینری، و تابع سوم برای خواندن مقدار مورد نیاز از فایل .txt داخل فایل نوشته شده اند:



همانطور که مشاهده میشود تابع read\_plaintext محتویات فایل را بازگردانده، سپس تابع encrypt\_plaintext آن را رمزنگاری کرده و در آخر تابع write\_ciphertext\_to\_file آن را در فایل ذخیره میکند.

**بخش ششم**

مطابق شکل زیر تو تابع برای decryption و خواندن مقدار رمزنگاری شده از فایل نوشته شده اند:



ابتدا تابع read\_ciphertext\_from\_file اجرا شده و به عنوان خروجی مقدار رمزنگاری شده را از فایل برمیگرداند. سپس تابع decrypt\_ciphertext آن را به رشته اصلی برمیگرداند.

**بخش هفتم و هشتم**

انجام شد.

**بخش نهم**

با اجرای فایل interactive.py میتوانید به صورت تعاملی با برنامه ارتباط برقرار کنید.

**نکات:**

* فایل رمز در .env ذخیره میشود.
* برای اجرا یکباره کل برنامه و توابع، فایل main.py را اجرا کرده و برای برنامه تعاملی، فایل interactive.py را اجرا کنید.