Zadanie 2 (Kryptografia)

September 30, 2024

Cieľom zadania je oboznámiť sa so základnými princípmi kryptografie ako sú symetrické/asymetrické šifry, digitálny podpis a ochrana integrity. V rámci tohto zadania budete dopĺňať funkcionalitu rôznych API volaní v jednoduchom webovom serveri. Kostra projektu je dostupná v súbore **zadanie_2.zip**. Samotný webserver je implementovaný v jazyku Python prostredníctvom knižníc *Flask* [2] (jednoduchý framework na implementáciu webservera) a *SQLAlchemy* [3] (rozšírenie pre Flask na podporu databáz). Knižnice je možné nainštalovať pomocou príkazu:

```
pip install flask flask-sqlalchemy
```

Po spustení programu bude webový server bežať na adrese **127.0.0.1** a porte **1337** (pozor, webový server neobsahuje žiadny frontend). Komunikovať s webovým serverom možno prostredníctvom programu *curl* [1]. Pomocou programu *curl* možno jednoducho posielať HTTP požiadavky na server. Jednotlivé API volania sú uvedené v kostre projektu. **Rozhranie pre API volania nemeňte**.

Úlohy:

1. Vyberte si kryptografickú knižnicu pre jazyk Python podľa svojho uváženia. V dokumentácii v stručnosti danú knižnicu popíšte, **zdôvodnite** prečo ste si ju vybrali a zdokumentujte jej inštaláciu.

2. Implementujte API volanie `/api/gen/<user>` (GET požiadavka). API volanie vygeneruje asymetrický kľúčový pár pre používateľa `user`. Verejná časť kľúča sa spolu s používateľským menom uloží do databázy a privátna časť kľúča sa odošle späť klientovi. **Výber asymetrickej šifry a jej parametrov zdôvodnite**. *curl* požiadavka, ktorá vygeneruje kľúčový pár pre používateľa `upb` a privátnu časť kľúča uloží na strane klienta do súboru `ubp.key` (kľúč musí byť v binárnej podobe):

   ```
   curl 127.0.0.1:1337/api/gen/ubp --output ubp.key
   ```

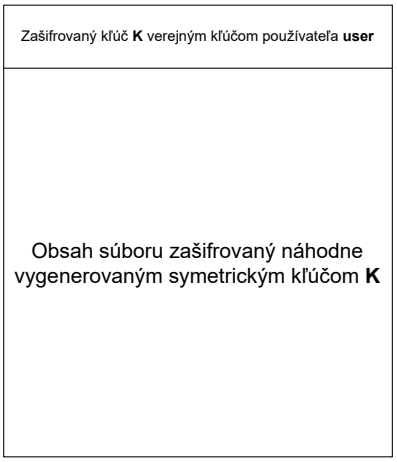

Figure 1: Ukážkový formát zašifrovaného súboru.

3. Implementujte API volanie `/api/encrypt/<user>` (POST požiadavka). Cieľom volania je zašifrovať súbor pre používateľa `user`, pričom je potrebné využiť kombináciu symetrickej a asymetrickej kryptografie. Klient posiela na server súbor, ktorý si želá zašifrovať spolu s používateľským menom, pre koho sa daný súbor šifruje. Po prijatí požiadavky server vygeneruje náhodný symetrický kľúč K, ktorým sa bude šifrovať obsah súboru. **Výber symetrickej šifry a jej parametrov zdôvodnite**. Po zašifrovaní obsahu súboru, sa symetrický kľúč K zašifruje verejným kľúčom K_V používateľa `user` (verejný kľúč sa načíta z databázy) a zašifrovaný kľúč sa spolu so zašifrovaným obsahom súboru odošle klientovi. Jednoduchú schému šifrovania a formátu výsledného súboru možno vidieť na obrázku 3. **Formát zašifrovaného súboru popíšte v dokumentácii**. Zašifrovaný súbor musí byť v binárnej podobe. *curl* požiadavka, ktorá zašifruje súbor `file.pdf`, pre používateľa `upb` a výsledok uloží do súboru `encrypted.bin`:

   ```
   curl -X POST 127.0.0.1:1337/api/encrypt/ubp -H "Content-Type:
   application/octet-stream" --data-binary @file.pdf
   --output encrypted.bin
   ```

4. Implementujte API volanie `/api/decrypt` (POST požiadavka). Volanie vykoná inverznú operáciu k predošlému volaniu. V tomto prípade musí klient okrem zašifrovaného súboru poskytnúť aj svoj privátny kľúč, ktorý ma lokálne u seba. *curl* požiadavka, ktorá dešifruje súbor `encrypted.bin` pomocou privátneho kľúča `upb.key` a výsledok (dešifrovaný obsah súboru) uloží do súboru `decrypted.pdf` (jedná sa o `multipart` požiadavku):

   ```
   curl -X POST 127.0.0.1:1337/api/decrypt -F "file=@encypted.bin"
   ```

```
-F "key=@ubp.key" --output decrypted.pdf
```

5. Implementujte API volanie `/api/sign` (POST požiadavka). Cieľom volania je vygenerovať digitálny podpis pre ľubovoľný vstupný súbor. V tomto prípade musí klient taktiež odoslať svoj privátny kľúč. *curl* požiadavka, ktorá digitálne podpíše súbor `document.pdf`, kľúčom `upb.key` a uloží digitálny podpis v binárnej podobe do súboru `signature.bin`:

   ```
   curl -X POST 127.0.0.1:1337/api/sign -F "file=@document.pdf" -F
   "key=@ubp.key" --output signature.bin
   ```

6. Implementujte API volanie `/api/verify/<user>` (POST požiadavka). Cieľom volania je overiť pravosť digitálneho podpisu pre používateľa `user`. Volanie vracia správu, či sa podarilo alebo nepodarilo verifikovať podpis (správa v JSON formáte). *curl* požiadavka, ktorá overuje digitálny podpis používateľa `upb` (verejný kľúč sa načítava z databázy), pre súbor `document.pdf` a digitálny podpis `signature.bin`:

   ```
   curl -X POST 127.0.0.1:1337/api/verify/upb -F "file=@document.pdf"
   -F "signature=@signature.bin" --output signature.bin
   ```

7. Implementujte API volania `/api/encrypt2/<user>` a `/api/decrypt2` (POST požiadavky). Jedná sa o totožné API volania ako v bodoch 3 a 4 s rozšírením o kontrolu integrity. V prípade ak zlyhá kontrola integrity pri dešifrovaní, vypíšte chybovú hlášku a v dešifrovaní nepokračujte. **Zdôvodnite a popíšte mechanizmus ochrany integrity**.

Deadline zadania je **15.10.2024** o **13:37**. Zadanie sa odovzdáva do **akademického informačného systému** (AIS) do zvoleného miesta odovzdania. Odovzdáva sa dokumentácia vo formáte **PDF** (!!!) a zdrojové kódy (iba .py súbory, databázu nie je nutné odovzdávať). Odovzdáva **jeden** člen z tímu.

Literatúra

[1] curl. `https://curl.se/`.

[2] Flask. `https://flask.palletsprojects.com/en/3.0.x/`.

[3] Flask-sqlalchemy. `https://flask-sqlalchemy.readthedocs.io/en/3.1.x/`.