**Password Manager**

Tri su moguća scenarija korištenja sustava:

1. [lokacija] init [lozinka] - pomoću lozinke [lozinka] koju je korisnik zadao kodira se string varijabla;
2. [lokacija] put [lozinka] [adresa] [lozinkaZaAdresu] – prvo se provjerava je li lozinka [lozinka] koju je korisnik unio ispravna. Ako je, onda se postupkom dekodiranja dekriptiraju podaci spremljeni na disku te se provjerava ima li unesena adresa neku postojeću lozinku – ukoliko ima, stara lozinka zamjenjuje se novom. Nakon toga, sve se ponovno kodira postupkom kodiranja;
3. [lokacija] get [lozinka] [adresa] – provjerava se ispravnost lozinke [lozinka] koju je korisnik unio. Ako je lozinka ispravna, podaci se dekodiraju postupkom dekodiranja, među njima se pronađe unesena adresa [adresa] i vrati se lozinka pohranjena za tu adresu.

U nastavku su opisani postupci kodiranja i dekodiranja podataka.

**Kodiranje podataka** započinje naredbom scrypt(). Njezin potpis je Crypto.Protocol.KDF.scrypt(password, salt, key\_len, N, r, p, num\_keys=1) i služi za derivaciju ključa. Parametri funkcije su:

* password – tajna šifra pomoću koje se generiraju ključevi;
* salt – nasumičan string korišten za bolju zaštitu;
* key\_len – duljina deriviranog ključa;
* N – cijena za CPU odnosno memoriju;
* r – veličina bloka;
* p – parametar paralelizacije;
* num\_keys – broj deriviranih ključeva.

Pomoću naredbe scrypt() generiraju se dva ključa. Jedan ključ koristi se za enkripciju podataka, a drugi za hash. Ovime je omogućena *encrypt-then-MAC* zaštita podataka, kojom se kasnije ostvaruje integritet i povjerljivost adresa i zaporki.

Pomoću prvog generiranog ključa stvara se novi objekt cipher uz pomoć MODE\_CBC parametra, što znači da će se za enkripciju podataka koristiti CBC algoritam. Tim objektom kriptiraju se podaci. Iz činjenice da se kriptira cijela datoteka proizlazi zaključak da je osigurana **povjerljivost** adresa i zaporki. Pri kreiranju objekta cipher stvoren je i inicijalizacijski vektor, koji potom pretvaramo u string i pohranjujemo na disk.

Pomoću kriptiranih podataka kreira se HMAC objekt hmac koji će služiti za provjeru ispravnosti glavne lozinke, ali i provjeru jesu li podaci bili mijenjani od treće strane, čime je osiguran **integritet** podataka.

Pozivanjem funkcije digest() nad objektom hmac dobije se hash generiran pomoću tog objekta i ranije generiranog hash ključa. Taj hash tag također se u obliku stringa pohranjuje na disk. Kodirani podaci pohranjuju se u zasebnu datoteku.

**Dekodiranje podataka** započinje dohvaćanjem inicijalizacijskog vektora, salta i taga s diska. Pomoću salta i upisane lozinke generiraju se dva ključa preko funkcije scrypt().

Kodirani podaci dohvaćaju se iz datoteke u kojoj su pohranjeni te se pomoću njih i drugog ključa generira HMAC objekt. Provjerava se ispravnost unesene lozinke uspoređivanjem tagova. Ako su dva taga ista, nastavlja se na dekodiranje podataka; ukoliko nisu, znači da je bilo pokušaja izmjene podataka ili je unesena lozinka netočna.

Nakon stvaranja novog AES objekta, pomoću funkcije decrypt() podaci se dekodiraju i uklanja se višak bajtova (*padding*).