**IZVJEŠTAJ SA 5. LABORATORIJSKE VJEŽBE**

Tijekom vježbe bavili smo se temom sigurnosti lozinki koristeći iterativno hashiranje, kreiranje soli i utjecaj različitih hash funkcija na vrijeme izvođenja. Cilj vježbe je razumjeti zašto ponekad koristimo "sporije" hash funkcije koje zahtijevaju više vremena za generiranje hasha i kako to utječe na sigurnost aplikacije. Vježba je podijeljena u dva dijela.

# DIO

U prvom dijelu vježbe provjerili smo koliko vremena neke kriptografske hash funkcije trebaju za hashiranje. Iako su sve funkcije brze, postoji značajna razlika u vremenu potrebnom za hashiranje informacije ili običnog teksta. Ove sitnice mogu se iskoristiti prilikom rada na aplikacijama s elementima prijave ili registracije korisnika. To se radi zbog sigurnosnog aspekta, jer ako napadač koristi rječnik za brute-force napad, svaki trenutak je važan. Usporavanjem hash funkcije i uvođenjem timeout-a (nakon nekoliko neuspjelih pokušaja lozinke morate pričekati određeno vrijeme) demotiviramo napadača. Naravno, korisnik želi da se prijava odvija neprimjetno, pa je važno postići balans između usporavanja napadača i vremena koje korisnik želi čekati za prijavu. Ukoliko radimo aplikaciju koja nije kritična za sigurnost, nećemo staviti toliku važnost na sigurnost, ali ako radimo aplikaciju koja sadrži važne podatke, korisnik će razumjeti ako prijava traje nešto dulje zato što je to u njegovom najboljem interesu. U prvom dijelu vježbe smo se bavili sigurnošću aplikacije. Ukoliko radimo aplikaciju koja nije kritična po pitanju sigurnosti, nećemo koristiti složene hash funkcije, međutim ako radimo aplikaciju koja sadrži važne podatke, korisnik će razumjeti ako proces prijave traje nešto dulje zato što je to u njegovom najboljem interesu. Kako bismo povećali sigurnost, neke hash funkcije koriste i sol koja služi kao faktor nepredvidljivosti. To nam omogućava da koristimo isti password na više mjesta jer imamo dodatnu zaštitu u obliku soli.

# DIO

U 2. dijelu vježbe smo kreirali SQLite bazu podataka za pohranu korisničkih imena i lozinki. Koristili smo getpass za unos lozinka kako ne bi bile prikazane na konzoli, stvorili smo bazu koja ne dopušta unos korisnika sa istim imenom, te provjeravamo ispravnost unosa username-a i lozinke kako bismo usporili napadača. Koristili smo SQL upite za provjeru postojanja korisnika i Argon2 funkciju za provjeru ispravnosti lozinke. Ako odgovor na SQL upit bude brz, napadač zna da je pogriješio u korisničkom imenu, a ako odgovor bude spor, napadač zna da je pogriješio u lozinci. Sve to smo koristili u kodu 2. dijela vježbe.