

Bezbednost u sistemima elektronskog poslovanja – Kontrolna tacka 1

Cuvanje podataka o korisnicima u bazi

Korisnik je jedinstveno identifikovan pomocu svoje email adrese. Da bi bilo koji korisnik pristupio funkcionalnostima aplikacije mora prethodno da se uloguje. Korisnik izvršava logovanje pomocu svog email-a i lozinke. Email je javno dostupna informacija i nema potrebe da je na bilo koji nacin dodatno osiguramo, zato je ona zapisana u bazu u obliku plain text-a. Medjutim, lozinka je privatna informacija i kompromitovanje lozinke bi imalo kritične bezbednosne posledice.

Pre cuvanja u bazu lozinke smo hesirali sa Bcrypt algoritmom. Iako je hes algoritam izuzetno težak “za razbiti” sve hes funkcije imaju jednu manu => iste ulaze transformisu u iste izlaze. Na primer, ukoliko dva korisnika odaberu iste lozinke, one ce u bazi biti skladistene kao isti hes i maliciozni napadac ce moci ukoliko sazna jednu lozinku, automatski da sazna i drugu. Za resavanje ovog problema Bcrypt algoritam koristi metod “salting hashes”.

“Salting hashes” na pocetak (prepending the salt) ili na kraj (appending the salt) svake lozinke dodaje neki slucajan niz karaktera.

Lozinka: 1234pass

Salt: f!nd!ngn3m0

“Salted” lozinka: 1234passf!nd!ngn3m0

Nakon kreiranja “salted” lozinke ona se hesira Bcrypt hes funkcijom.

Izuzetno je bitno da svaki korisnik u nasem sistemu ima jedinstven “salt”. Na primer: Ukoliko bismo uvek isti salt dodavali na kraj svake lozinke i ako bi imali dva korisnika koja odaberu istu lozinku. Dodavanjem istog salt-a samo bismo kreirali duze lozinke koje ne bi bile jedinstvene u nasem sistemu jer bi obe “salted” lozinke rezultovale istim hesom. Medjutim, ukoliko bi svaki korisnik imao jedinstven salt dobili bismo jedinstvene i duze lozinke koje imaju drugacije heseve.

Bcrypt koristi nasumicno izgenerisanu salt vrednost od 16 bajtova.

Jedan od boljih clanaka iz kog smo naucili dosta o hesiranju lozinki pomocu salt-a je:

<https://auth0.com/blog/adding-salt-to-hashing-a-better-way-to-store-passwords/>

Ostali linkovi:

https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html#salting,

<https://docs.spring.io/spring-security/site/docs/5.0.19.RELEASE/reference/htmlsingle/> - 37.4

poglavlje BcryptPasswordEncoder

<https://stackoverflow.com/questions/6832445/how-can-bcrypt-have-built-in-salts> - nacin na koji je “salting” mehanizam ugradjen u Bcrypt

Spring Security i Jwt Token

Spring Security je implementiran u obe aplikacije. REST sam po sebi ne podrzava pracenje sesije, a mi u nasoj aplikaciji zelimo da znamo koji klijent salje zahteve sa fronta. Za prenosenje informacija o korisniku odlucili smo se za upotrebu JWT tokena. JWT token se sastoji iz tri dela: header-a, payload-a i signature. U header (zaglavlju) se nalaze informacije o tipu tokena – u nasem slucaju JWT token, i tipu algoritma koji je koriscen za sifrovanje. Payload sadrzi sve podatke koje mi zelimo da prosledimo – podaci o korisniku, njegove authorities (uloge na osnovu kojih ima pristup odredjenim funkcionalnostima), datum izdavanja JWT tokena... Hesirani (mi smo koristili HS512 za hesiranje) podaci iz header-a i payload-a se potpisuju tajnim kljucem koji se nalazi na serveru. Ukoliko neko uzme nas JWT token i izmeni njegove podatke zbog nemogucnosti pristupa tajnom kljucu servera nece moci ponovo da ih potpise i znace se da su podaci kompromitovani.

Prilikom logovanja na aplikaciju klijent dobija svoj JWT token - autentifikacija. Svaki naredni zahtev koji korisnik salje mora u sebi da sadrzi JWT token (interceptors u Angular aplikaciji) – autorizacija.

Angular aplikacija

Kao nacin zastite pristupa u Angular aplikaciji smo implementirali Guards. Guards su postavljeni na sve rute, osim na login rutu kojoj svi imaju pristup. Guard pri svakom pristupu nekoj ruti proverava JWT token trenutno ulogovanog korisnika, i ukoliko korisnik ima ulogu kojoj je dozvoljen pristup zahtevanoj ruti Guard ga “pusta” – dozvoljava mu redirektovanje na rutu. Medjutim, ako mu nije dozvoljen pristup trazenoj ruti preusmeren je na login stranicu, i ima samo mogucnost logovanja u aplikaciju.

Kreiranje zahteva za sertifikat (CSR)

Ukoliko admin bolnice zeli da njegova bolnica ima sertifikat mora da isprati sledece korake:

- 1) Uloguje se na hospitalApp pomocu svog email-a i lozinke
- 2) Odabere “Create Certificate Signing Request”
- 3) Popunjava formu informacijama o sebi i bolnici
- 4) Nakon unosenja informacija kreira se par kljucева – privatni i javni kljuc. Privatni kljuc se cuva u fajlu, a javni kljuc se zajedno sa unetim informacijama konvertuje u csr
- 5) hospitalApp salje csr adminApp


(nije bilo potrebe da se csr na neki nacin zastiti pre slanja, jer ne sadrzi nikakve tajne informacije)

- 6) adminApp prima csr. Kao aplikacija od poverenja, admin ne moze bilo kome da izdaje sertifikate, vec mora da proveri da li je osoba koja je poslala csr zaista ona za koju se izdaje. Mi smo se odlucili da ce admin od kreatora csr-a zahtevati da dokaze "kontrolu nad domenom" tako sto ce mu poslati email sa verifikacionim linkom na email adresu iz csr-a. => Admin salje email
("Domain control": <https://www.ssllabs.com/knowledgebase/how-can-i-complete-the-domain-control-validation-for-my-ssl-certificate/>)
- 7) Admin iz hospitalApp klikom na verifikacioni link iz email-a potvrđuje "kontrolu nad domenom" i njegov csr postaje validiran (tek nakon sto je csr validiram postaje vidljiv u adminApp).


Pregled zahteva za sertifikat

- 1) Admin se uloguje na adminApp pomocu svog email-a i lozinke
- 2) Odabere "Certificate Signing Requests"
- 3) Prikaze mu se tabelarni pregled informacija o svim pristiglim csr-ovima

Odbijanje zahteva za sertifikat(CSR)

- 1) Admin se uloguje na adminApp pomocu svog email-a i lozinke
- 2) Odabere "Certificate Signing Requests"
- 3) Klikom na  pored odgovarajuceg csr-a u tabeli => csr se odbija i obavestenje se salje na mejl

Kreiranje sertifikata

- 1) Admin se uloguje na adminApp pomocu svog email-a i lozinke
- 2) Odabere "Certificate Signing Requests"
- 3) Klikom na  pored odgovarajuceg csr-a u tabeli => redirekcija na kreiranje sertifikata
- 4) Prvi korak u kreiranju sertifikata je detaljan pregled informacija o csr-u (izmena podataka nije dozvoljena)
- 5) Drugi korak je odabir dodatnih podataka o sertifikatu – template sertifikata (tls_server, tls_client, ca_cert, end_user) i odabir datuma vazenja sertifikata (templateji imaju odredjene KeyUsage-e)
(KeyUsage explained:
https://help.hcltechsw.com/domino/10.0.1/admin/conf_keyusageextensionsandextendedkeyusage_r.html)

- 6) Treci korak je odabir izdavaca sertifikata. U padajucem meniju su prikazani svi dozvoljeni izdavaci sertifikata. Nakon odabira izdavaca automatski se popunjava detaljan prikaz informacija o izdavacu (izmena podataka nije dozvoljena)
- 7) Klikom na dugme "Create" se kreira sertifikat i salje se email traziocu sertifikata sa linkom na koji klikom skida sertifikat u .crt formatu.

Odlucili smo se za ovaj nacin distribucije sertifikata jer nam je pored fizickog prenosa sertifikata (na primer skinemo sertifikat na usb i odnesemo do odgovarajuce masine na kojoj treba da se instalira sertifikat) bio najbezbedniji – ako admin bas zeli na taj nacin da distribuira sertifikat iz detaljnog pregleda sertifikata mu je omoguceno skidanje sertifikata. Takodje, slanje na email nam pruza sigurnost da je sertifikat dostavljen korisniku koji ga je i zatrazio. Naravno, da bi ovo zaista bilo bezbedno mora da se omoguci https komunikacija – bice uradjeno u nekoj od narednih etapa kreiranja aplikacije

Pregled svih sertifikata

- 1) Admin se uloguje na adminApp pomocu svog email-a i lozinke
- 2) Odabere "Certificates"
- 3) Otvara se prikaz sertifikata u obliku stabla – klikom na sertifikat se otvaraju svi sertifikati koje je on izdao/potpisao

Detaljan pregled sertifikata i skidanje sertifikata

- 1) Admin se uloguje na adminApp pomocu svog email-a i lozinke
- 2) Odabere "Certificates"
- 3) Navigacijom kroz stablo sertifikata do zeljenog sertifikata i klikom na dugme "Detailed" otvara se dijalog sa detaljnim pregledom sertifikata
- 4) Klikom na dugme "Download" se skida sertifikat u .crt formatu

Povlacenje sertifikata

- 1) Admin se uloguje na adminApp pomocu svog email-a i lozinke
- 2) Odabere "Certificates"
- 3) Navigacijom kroz stablo sertifikata do zeljenog sertifikata i klikom na dugme "Revoke" otvara se dijalog sa predefinisanim razlozima za povlacenje sertifikata (<https://security.stackexchange.com/questions/174327/definitions-for-crl-reasons>)
- 4) Odabirom razloga za povlacenje sertifikata i klikom na dugme "Revoke" sertifikat se povlaci i email se salje vlasniku sertifikata sa informacijom o razlogu povlacenja sertifikata

Instalacija sertifikata

Nakon sto je admin hospitalApp skinuo svoj sertifikat potrebno je da ga instalira na svojoj masini. Za sada je moguće samo lokalno instaliranje na masini, ali nakon implementacije https-a će biti omogućena i instalacija sertifikata u browseru.

Postoji programsko i manuelno instaliranje sertifikata na lokalnoj masini => mi smo se odlucili za manuelno instaliranje. Postoji više načina da se sertifikat instalira.

Prvi način: dvoklikom na sertifikat pojavljuje se prozor gde ima ponudjeno dugme "install certificate" odaberemo Trusted Root Certification Authorities za store i kliknemo Finish

Drugi način: desni klik na Start, odaberemo Run, ukucamo 'mmc' i pritisnemo 'Ok'. Na 'User Account Control' prozoru odaberemo 'Yes'. Kada se 'Microsoft Management Console' pojavi kliknemo 'File' i odaberemo 'Add/Remove Snap-in', u meniju odaberemo 'Certificates' i kliknemo 'Add'. Na sledecem prozoru odaberemo 'Computer account', kliknemo Finish pa Ok. Na 'Microsoft Management Console' prozoru kliknemo "Certificates (Local Computer)" dvoklik na 'Trusted Root Certificate Authorities' i desni klik na 'Certificates' unutar tog foldera i izaberemo 'All tasks' pa 'Import...'. Pretražimo sertifikat do direktorijuma u kojem smo ga sacuvali i u 'Certificate Store' obavezno odaberemo 'Trusted Root Certificate Authorities' i kliknemo Next pa Finish (Link: <https://support.securly.com/hc/en-us/articles/360026808753-How-do-I-manually-install-the-Securly-SSL-certificate-on-Windows?fbclid=IwAR2eO-QQa7NZcOlal30tY7ggBT3HIG3gO11J0To7hlhyTL4UvLCo-PchpM>)

Treci način: pokrenemo komandnu liniju u direktorijumu u kojem je sacuvan sertifikat i unesemo komadu 'keytool -import -alias myalias -file example.crt -keystore exampleraystore' gde je alias nas interni a example.crt naziv sertifikata sa ekstenzijom (https://docs.oracle.com/javase/tutorial/security/toolsign/rstep2.html?fbclid=IwAR1KN4A4YevTz7y0qju9J6AMC_kNQzoghWHP5tLI0iv_on3BpPKsTe1MKd4)

Zamena isteklog sertifikata

Kada sertifikat istekne, potrebno je obnoviti ga. Obnova sertifikata se ne razlikuje puno od kupovine novog. Ukoliko zaboravimo da obnovimo nas sertifikat u browser-u će se prikazati pop-up prozor sa porukom "The site's security certificate has expired" svakom klijentu koji poseti nasu aplikaciju i transakcije više nisu bezbedne. Postoji odredjeni period u kom sertifikat može da se obnovi, a to je do 120 dana pre isteka i 30 dana nakon isteka sertifikata.

Proces obnove sertifikata je:

- 1) Generisemo CSR
- 2) Selektujemo sertifikat
- 3) Podesimo period vazenja sertifikata (obicno 1 ili 2 godine)
- 4) Popunimo sve neophodne podatke

5) Platimo i izvršimo deployment na nas server

Treba da obratimo paznju na to da svi podaci prilikom obnove sertifikata treba da budu revalidirani. Tj treba da se odradi DCV (Domain Control Validation) npr putem mejla. HTTPS ili DNS validacija koja je odabrana u procesu aktivacije sertifikata. Ukoliko imamo OV (Organization validation) ili EV (Extended Validation) sertifikat, sve informacije kompanije ce se verifikovati ponovo i neophodni dokumenti ce biti ponovo poslani CA-u.

Sertifikat ce posle uspesne obnove biti prosledjen email-om, gde je posle neophodno da ga instaliramo na nas server.

Svaki put kada obnavljamo sertifikat potrebno je da generisemo novi CSR/RSA par i da sacuvamo nas RSA, neophodno je da koristimo iste kredencijale kao pri proslom generisanju CSR-a.

U zavisnosti od sertifikata, postoje drugaciji vremenski periodi za obnovu (kupovinu):

- za Domain Validation (DV) SSL Certificate je potrebno 1 do 2 sata
- za Organization Validation (OV) SSL Certificate 4 do 5 dana
- za Extended Validation (EV) SSL Certificate 1 do 2 nedelje

Kratak period vazenja sertifikata nam pomaze da ublazimo narusavanje bezbednosti kljuceva, jer se svaki put generisu novi, takodje kako bi bili sigurni da svi sertifikati koriste najnovije standarde za bezbednost. (Shorter life certificate helps in mitigate compromises of keys, as new keys are generated every time. Also, to ensure all certificates are using the latest security standards.)

Just like a government ID or passport, it's important that the information is occasionally rechecked. Imagine if you never had to get a passport or driver's license renewed! Many people would still be carrying around IDs with a picture of them from when they were sixteen and much shorter. Using those IDs for accurate identification would be much harder and far less reliable.

Postoje alati koji salju podsetnike pred istek sertifikata, takodje u zavisnosti od toga gde je sertifikat kupljen, postoje kompanije koje same obavestavaju klijente.

Linkovi: <https://www.thesslstore.com/blog/ssl-certificates-expire/?fbclid=IwAR3XHhbTQijRc9Rn8IBVfdZ5I2O05jVey90fyXtA8RX4C1VKkkSjW88yJV4>
https://www.https.in/ssl-security/renew-ssl-certificate-steps/?fbclid=IwAR2mC_fexzAQV2lIEzS5QAwawqZBdbhU-a4kypz74rDSh7FGp8nk2oREJhQ