Elevation of privilege

**Narušavanje povlastica**(*elevation of privilege* ili *privilege escalation*) je akt koji se zasniva na eksploataciji nekog *bug-a,* rupe u dizajnu ili previdu u konfiguraciji operatvinog sistema ili aplikacije, da bi se doslo do resursa koji su u normalnim okolnostima nedostupne nekim od korisnika ili aplikacijama.

Gore navedeno za posledicu ima da se stvore privilegije iste ili veće od autorizovanih sistemskih admina ili projektanta, čime se mogu izvršavati nedozvoljene akcije.

Narušavanje povlastica se javlja u dva oblika:

1. **Vertikalno povećanje povlastica**(*Vertical privilege escalation* ili *privilege elevation*) predstavlja situaciju kada korisnik ili aplikacija sa manje povlastica pristupa funkcijama ili sadržaju koji su rezervisani za korisnike ili aplikaciju sa većom povlasticom.

Postoje razni primeri ovakvih situacija.

Glavna ideja se zasniva na tome da aplikacija sa nivoima pristupa često predpostavlja da će uvek biti snabdevena informacijama koje odgovaraju njenoj specifikaciji i često se te iste informacije ne proveravaju, odnosno potvrđuju.

*1.1. XSS* ili *Cross-Site-Scripting* kod koga je ideja da napadač obori sigurnosni model web pretraživača ili zaobiđe kontrolu pristupa(najčešće je to *polisa zajedničkog porekla* ili [*same-origin policy*](https://en.wikipedia.org/wiki/Same-origin_policy), koja onemogućava da neka druga web stranica pristupa [*DOM stablu*](https://en.wikipedia.org/wiki/Document_Object_Model) prve stranice) tako da može da ubaci zarazan kod([*client-side-script*](https://en.wikipedia.org/wiki/Client-side_scripting)*)* koji može da se izvršava na korisnikovom web pretraživaču i time dovede do gore navedene situacije(pristupanje i manipulacija resursima koji nisu dozvoljeni za korisnike sa manjom povlasticom).

1.2. [*Shell injection*](https://en.wikipedia.org/wiki/Code_injection)se zasniva na ideji da se ne potvrđena informacija prosledi kao izvršna komanda, koja će dovesti do mogućnosti napadača da manipuliše sa resursima ili akcijama za koje on nema direktna prava.

1.3. Često se koristi i termin *Jailbreaking* koji se odnosi na sposobnost da se ‘pobegne iz zatvora’, gde zatvor predstavljaju razne kontrole pristupa, kako operativnog sistema, tako i ostalih aplikacija.

Kao primer za ovakav slučaj u našoj aplikaciji se može navesti situacija da običan građanin, dakle ne odbornik ili predsednik skupštine, može da pristupi resursima, za koje on nema odobrena prava pristupa(usvajanje akata, amandmana, povlačenje akata ili amandmana, kao i samo predlaganje istih).

Mogućih strategija za ublažavanje ovakvih problema ima nekoliko.

* [*Principle of least privilege*](https://en.wikipedia.org/wiki/Principle_of_least_privilege) kod koga je ideja da svaki modul (korisnici, procesi, sama aplikacija) mogu da koriste samo one resurse koji su im potrebni za normalan rad i funkcionisanje.
* Omogućiti da se odradti validacija, kako na klijentskoj strani, tako i na serverskoj, odnosno potvrda samih informacija i resursa koji se primaju i koji se šalju(*Shell injection*).
* Onemogućiti korišćenje skripti web pretraživača pre korišćenja aplikacije. Neke aplikacije su tako napisane da omogućavaju funkcionisanje bez korišćenja bilo kakvih *client-side-scripts*.
* *WEB application firewall* koji postavlja određena pravila u HTTP konverzaciji. Na ovaj način, pravilo konfigurisan, mogu se prepoznati i blokirati XSS napadi. Čak se ide i korak dalje, omogućavajući da se prepoznaju novi napadi, tako što se uočavaju nepoznati šabloni.

1. **Horizontalno povećanje povlastica**(Horizontal privilege escalation) predstavlja situaciju kada običan korisnik pristupa funkcijama ili sadržaju nekog drugog običnog korisnika.

Često je mnogo lakše pristupiti resursima drugog običnog korisnika, nego samim sistemskim, odnosno pravima pristupa admina.

Postoji nekoliko razloga za to, a onaj glavni je ljudska bezobzirnost, nezainteresovanost ili obična glupost.

Mogu se potrošiti ogromne svote novca ili razviti genijalne ideje, ali ljudski faktor ipak i dalje najviše utiče na celokupan proces.

* Predvidive, tj. očigledne i lako pogodljive lozinke(lozinka, password, 123456789, qwerty, asdfgh itd. itd.)
* Predvidivi ID sesije u HTTP Cookie-ju.
* Sama krađa HTTP Cookie-ja.
* ID sesije koji je vidljiv u URL-u.
* *Session fixation* gde se nakon krađe sesije drugog korisnika, postavlja ista u web aplikaciju, čime se dobije pristup resursa tog korisnika.
* Kao primer u našoj aplikaciji, može se navesti situacija da drugi odbornik na neki način preuzme kontrolu nad resursima drugog odbornika ili možda čak i predsednika skupštine, čime dobija dodatne mogućnosti za manipluaciju.

Mogućih strategija za ublažavanje ovakvih problema ima nekoliko.

* Prilikom kreiranja samih lozinka, biti barem malo maštovit, upotrebiti neka od [pravila](http://www.howtogeek.com/195430/how-to-create-a-strong-password-and-remember-it/) za kreiranje lozinki, kao i samo [merenje](https://howsecureismypassword.net/) jačine lozinke.
* Koristiti TLS prilikom HTTP komunikacije.
* Istražiti postojeća i proverena rešenja