

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

SEMINARSKI RAD 1662

UPOTREBA ALATA ZAP KAO ZAMJENA ZA ALAT BURP SUITE PROFESSIONAL

Ante Čavar

Voditelj: prof.dr.sc. Stjepan Groš

Zagreb, lipanj, 2024.

Zahvaljujem se prof. dr. sc. Stjepanu Grošu na pruženoj pomoći prilikom izrade rada.

Sadržaj

1. Uvod	1
2. Funkcionalnosti	2
2.1. OWASP ZAP	2
2.1.1. Presretački <i>proxy</i>	3
2.1.2. Aktivno i pasivno skeniranje	3
2.1.3. Skeniranje portova	4
2.1.4. Aplikacijsko programsko sučelje	4
2.1.5. <i>Fuzzer</i>	5
2.1.6. Trgovina ekstenzija (engl. <i>Marketplace</i>)	5
2.2. Burp Suite	6
2.2.1. <i>Proxy</i>	6
2.2.2. <i>Intruder</i>	6
2.2.3. <i>Repeater</i>	7
2.2.4. <i>Collaborator</i>	7
2.2.5. <i>Comparer</i>	7
2.2.6. <i>Extensions</i>	7
2.3. Usporedba alata <i>a priori</i>	8
3. Testiranje	9
3.1. Vidljiva pogreška bazirana na SQL injekciji	9
3.1.1. Vježba 1: Burp Suite	10
3.1.2. Vježba 1: ZAP	13
3.2. Dvofaktorska autentifikacija sa pogrešno konfiguriranom logikom	14
3.2.1. Testiranje: Burp Suite	14

3.2.2. Testiranje: ZAP	16
3.3. Napad silom na lozinku koristeći lošu implementaciju promjene lozinke . .	17
3.3.1. Testiranje: ZAP	18
3.3.2. Testiranje: Burp Suite	21
4. Usporedba i nadomještanje funkcionalnosti	22
5. Zaključak	24
6. Literatura	25
Sažetak	26
Abstract	27
Skraćenice	29

1. Uvod

Sigurnost web aplikacija je bitan dio razvoja moderne programske potpore. Kako se napadi na web aplikacije povećavaju iz dana u dan, potrebno je osim razumijevanja prijetnji i ranjivosti, imati i alate kojima ranjivosti a s njima i prijetnje možemo svesti na prihvatljivi minimum.

U ovom radu će se usporediti dva najraširenija alata za skeniranje i testiranje sigurnosti web aplikacija: OWASP ZAP i Burp Suite. Cilj je identificirati razlike u funkcionalnostima između ova dva alata, istražiti kako se mogu nadomjestiti funkcionalnosti koje ZAP nema, te opisati kako se pišu dodaci i proširuju mogućnosti ZAP-a. Na kraju, testirat će se nekoliko laboratorijskih vježbi koje se inače rješavaju pomoću alata Burp suite, koristeći ZAP.

Burp Suite je moćan, ali komercijalan alat koji zahtijeva skupu licencu za pristup svim funkcionalnostima. Za potrebe istraživanja i učenja, ovaj trošak može biti značajan. S druge strane, OWASP ZAP je alat otvorenog koda koji je besplatan za korištenje, ali potencijalno slabijih mogućnosti. Usporedbom ova dva alata, može se procijeniti koliko ZAP može biti održiva zamjena za Burp Suite u raznim scenarijima.

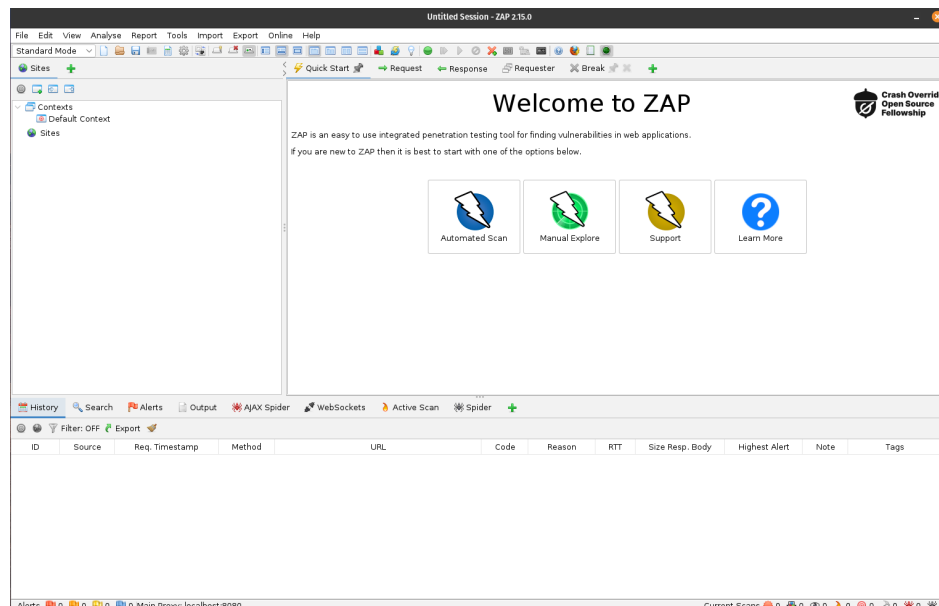
Najprije će se opisati osnovne funkcionalnosti oba alata. Zatim će se provesti nekoliko testova na različitim sigurnosnim scenarijima koristeći oba alata. Nakon toga, analizirat će se rezultati testova i usporediti performanse i mogućnosti ZAP-a i Burp Suite-a. Konačno, istražiti će se mogućnosti proširenja funkcionalnosti ZAP-a kroz dodatke i skripte te predložiti rješenja za funkcionalnosti koje Burp Suite ima, a ZAP nema i pritom objasniti koji alat odabrati i zašto.

2. Funkcionalnosti

Kako oba alata imaju pregršt funkcionalnosti navesti ću njihove najzanimljivije i najbolje implementirane funkcionalnosti koje će kasnije biti uspoređene.

2.1. OWASP ZAP

OWASP ZAP (još znan i kao Zed Attack Proxy) je program otvorenog koda namijenjen za skeniranje web aplikacija tj. stranica. Produkt je organizacije *Open Web Application Security Project* (OWASP) te održavan kako od njih tako i od nemale skupine programera i sigurnosnih stručnjaka iz cijelog svijeta. Alat pruža široki skup mogućnosti za detekciju i testiranje ranjivosti u web aplikacijama. Izgled ZAPa pri njegovom pokretanju vidljiv je na slici 2.1.



Slika 2.1. Izgled OWASP ZAP-a pri pokretanju

2.1.1. Presretački proxy

Intercepting Proxy, poznat i kao "*Break*" u *OWASP ZAP*-u, ključna je funkcionalnost koja omogućava korisnicima da presreću i pregledavaju *HTTP/HTTPS* zahtjeve i odgovore između preglednika i web aplikacije. Ovo omogućuje korisnicima da detaljno pregledaju i modificiraju podatke prije nego što su oni poslani na server ili primljeni u pregledniku. Presretanje prometa korisno je za ručno testiranje sigurnosti jer omogućuje detaljnu analizu *HTTP* zahtjeva i odgovora, uključujući zaglavlja, kolačiće, tijela zahtjeva i odgovore. Prije nego što se zahtjev pošalje ili odgovor primi, korisnici mogu ručno promijeniti podatke kako bi testirali različite scenarije.

Integrirani *Heads-Up Display (HUD)* omogućava vizualno praćenje i kontrolu presretanja prometa direktno iz preglednika. Korisnici mogu simulirati razne vrste napada presretanjem i modificiranjem prometa, kao što su *SQL* injekcije, *XSS* napadi, *CSRF*, i druge. [1, 2]

2.1.2. Aktivno i pasivno skeniranje

Aktivno skeniranje (*Active scan*) je ispitivanje ranjivosti pri kojem ZAP aktivno interaktira s aplikacijom ne bi li otkrila sve potencijalne sigurnosne prijetnje. Skeniranje uključuje slanje zahtjeva sistemu koji se testira (eng. *system under test*, nadalje SUT) te analiziranje odgovora ne bi li dobili uvid u ranjivosti aplikacije.

Pasivno skeniranje (*Passive scan*) je tip skeniranja gdje ZAP ne interaktira direktno s ijednim dijelom sustava na ijedan način. Kako bi se pasivno prikupljale informacije, alat nadgleda mrežni promet za potencijalne ranjivosti. U pozadini alat i dalje nadgleda zahtjeve i odgovore no ne zamarajući korisnika s njima sve dok ne otkrije problem, onda podigne *alert*.

Dobra stvar kod skeniranja u ZAP-u je mogućnost kontrole politike skeniranja. Ona nam omogućava da alatu pomognemo da bolje, lakše i brže pronađe ranjivosti koristeći znanje koje mi trenutno imamo, mogućnosti hardwera s kojim testiramo te vrste skenera koje imamo na raspolaganju. Te se politike mogu spremati te kasnije koristiti kao predložak u budućem testiranju. [1, 2]

2.1.3. Skeniranje portova

Funkcionalnost skeniranja portova u *OWASP ZAP*-u obavlja skeniranje ranjivih portova nad *SUT*-om kako bi utvrdio ranjive portove. U svojoj suštini radi isto što i određeni testovi za *Nmap* [3], pronalazeći otvorene i zatvorene portove. Također, kao i *Nmap*, može pronaći koji je trenutni servis na nekom od portova.

Služi kako bi se dobile dodatne informacije o potencijalnim vektorima napada. Pored toga, *OWASP ZAP* može integrirati rezultate skeniranja portova sa svojim ostalim alatima za sigurnosno testiranje, omogućujući sveobuhvatnu analizu sigurnosti.

Za korištenje ove funkcionalnosti, potrebno je omogućiti *Port Scanner* dodatak u *OWASP ZAP*-u. Nakon omogućavanja, korisnici mogu konfigurirati skeniranje da cilja specifične portove ili opsege portova, prilagoditi vrijeme čekanja na odgovore, i odabrati protokole nad kojima će se testirati. [2]

2.1.4. Aplikacijsko programsko sučelje

API omogućava korisnicima programski pristup direktno kodu tj. njegovoj funkcionalnosti. *API* omogućava automatizaciju određenih aspekata web testiranja te integraciju s ostalim alatima.

Proširivost *ZAP*-a je upravo ono što ga čini toliko moćnim naspram drugih alata. *ZAP API* podržava više jezika, uključujući *Python*, *Javu*, *JavaScript*, *Ruby*, i druge, što omogućava široku primjenu u različitim okruženjima. Kroz *API*, korisnici mogu pokretati skeniranja, pristupati rezultatima, mijenjati postavke i čak kreirati prilagođene napade. *ZAP API* se može koristiti za kontinuiranu integraciju (*CI*) i kontinuiranu isporuku (*CD*), omogućavajući sigurnosno testiranje kao dio razvojnih procesa. Integracija s alatima kao što su *Jenkins* i *GitLab CI* omogućava automatsko pokretanje sigurnosnih skeniranja prilikom svakog *build*-a, čime se osigurava otkrivanje potencijalnih ranjivosti ranije u procesu. Također, *ZAP API* omogućava jednostavno skaliranje testiranja. Korisnici mogu pokretati paralelna skeniranja na različitim ciljevima, što ubrzava proces testiranja velikih aplikacija. *API* podržava sve glavne funkcionalnosti *ZAP*-a, uključujući *Spider*, *Active Scan*, *Passive Scan*, *Forced Browse*, i druge, čime se omogućava sveobuhvatno testiranje kroz automatizirane skripte.

Dokumentacija za *ZAP API* je detaljna i pruža primjere za korištenje u različitim programskim jezicima, što olakšava integraciju i korištenje čak i za one koji nisu stručnjaci za sigurnost. [2]

2.1.5. Fuzzer

OWASP ZAP fuzzer stvara jedinstvene zahtjeve (engl. *payloads*) koje potom šalje na *SUT* koristeći već postojeće zahtjeve kao početnu odnosno referentnu točku. Fuzzer ima četiri načina rada: *safe*, *protected*, *standard*, i *ATTACK*. Svaki od navedenih načina pruža drukčije funkcionalnosti te načine rada. *OWASP ZAP fuzzer* također omogućava korisnicima da prilagode i kreiraju vlastite zahtjeve za specifične testove. Korisnici mogu definirati različite uzorke, vrijednosti i sekvence koje će se koristiti u *fuzzing* procesu. *Fuzzer* je integriran sa ostalim alatima u *ZAP*-u, omogućujući kombiniranje rezultata i sveobuhvatnu analizu. *ZAP fuzzer* pruža detaljne izvještaje o rezultatima fuzzing testa, uključujući sve pronađene ranjivosti, odgovore servera, i potencijalne sigurnosne probleme. Ova funkcionalnost je ključna za dubinsko testiranje i identifikaciju skrivenih ranjivosti u web aplikacijama. [1]

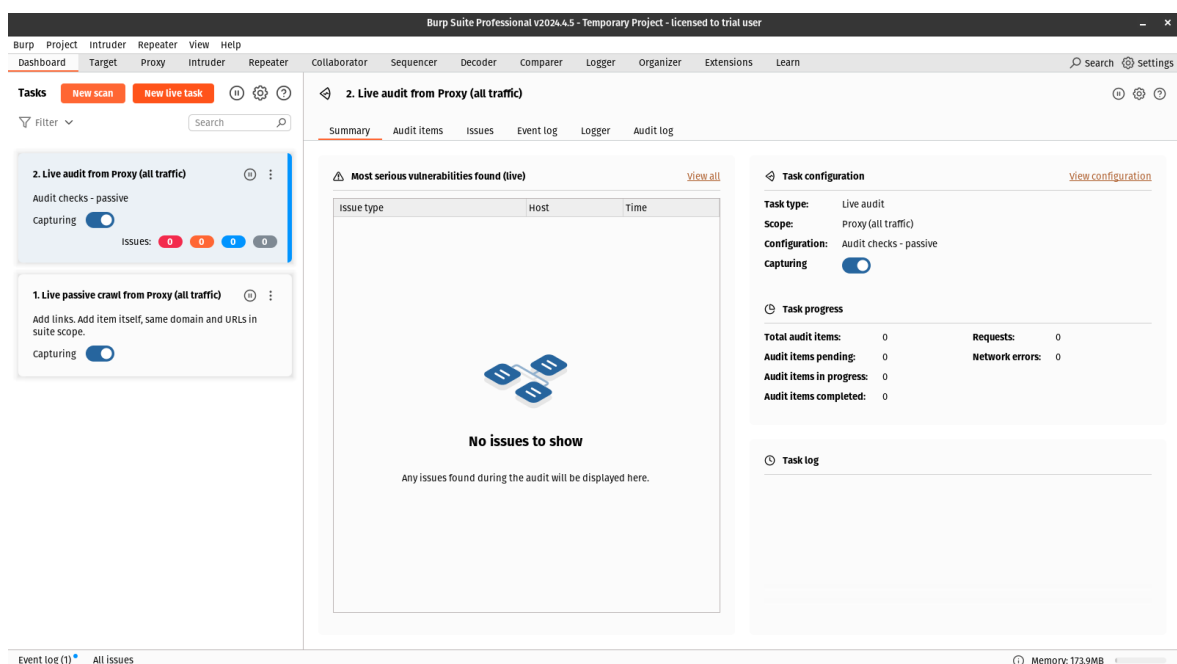
2.1.6. Trgovina ekstenzija (engl. Marketplace)

Marketplace, iako nije direktno povezan s web sigurnošću, predstavlja vrlo važan dio *ZAP*-a. Omogućuje korisnicima proširenje funkcionalnosti *ZAP*-a putem ekstenzija koje su razvili programeri iz cijelog svijeta. Sustav ocjenjivanja i metrike preuzimanja osiguravaju preporuku najkvalitetnijih ekstenzija.

Upravo zbog ove funkcionalnosti je *ZAP* uspio dobiti reputaciju kakvu trenutno ima. Zbog jednostavnosti modificiranja te principa otvorenog koda ambiciozna zajednica ljudi se brzo okupila kako bi pretvorili ovaj besplatan alat u sveobuhvatnog suradnika pri testiranju web aplikacija. [2]

2.2. Burp Suite

Uz ZAP postoji Burp Suite, stariji i komercijalni alat namijenjen za sigurnosno testiranje web aplikacija. Burp suite je razvila kompanija PortSwigger koja se također brine za nje-govo održavanje. Burp suite omogućuje pristup velikom rasponu alata, od skenera za traženje ranjivosti pa sve do naprednih alata za *fuzzing* i analizu web aplikacija. Početni prozor pro-grama možemo vidjeti na slici 2.2. Ovdje su pri vrhu vidljive kartice za određene module, lijevo se nalaze takozvani *live taskovi*, u sredini su pronađene ranjivosti, a desno konfigura-cija *live taskova*.



Slika 2.2. Izgled Burp suitea pri pokretanju

2.2.1. Proxy

Burp *Proxy* je jedan od glavnih modula Burp Suitea. On omogućava presretanje, pregled i promjenu HTTP i HTTPS zahtjeva između preglednika i web aplikacije. Koristeći Proxy, sigurnosni stručnjaci mogu analizirati zahtjeve i odgovore, tražiti ranjivosti i manipulirati podacima kako bi testirali sigurnost web aplikacije. [4]

2.2.2. Intruder

Burp *Intruder* je modul za automatizirane napade. Može se koristiti za *brute force* napade, testiranje valjanosti unosa, traženje skrivenih direktorija ili datoteka, te druge vrste napada.

Korisnik može konfigurirati različite parametre i payload-ove koji će se automatski slati, a Intruder će analizirati odgovore kako bi identificirao potencijalne ranjivosti. [4]

2.2.3. Repeater

Repeater omogućava korisnicima da ponavljaju HTTP/HTTPS zahtjeve ručno. Ovo je korisno za detaljno testiranje specifičnih zahtjeva i odgovora. Korisnik može modificirati i ponovo poslati zahtjeve, te analizirati odgovore kako bi identificirao ranjivosti i prijetnje. [4]

2.2.4. Collaborator

Collaborator je modul za otkrivanje server-side ranjivosti koje zahtijevaju interakciju s vanjskim sustavima/stranicama. Primjeri ovih ranjivosti uključuju SSRF (engl. *Server-Side Request Forgery*) i različite vrste injekcija koje rezultiraju vanjskim interakcijama. Omogućava postavljanje posebnog servera koji prati ove interakcije i pomaže u identifikaciji ranjivosti. [4]

2.2.5. Comparer

Comparer omogućava uspoređivanje dva skupa podataka. Ovo može biti korisno za analizu promjena između dva odgovora ili zahtjeva, kako bi se identificirale suptilne razlike koje mogu ukazivati na ranjivosti kao što su neodgovorna obrada zahtjeva pri kojoj mogu iscuriti informacije iz takozvanih 'sporednih kanala'. [4]

2.2.6. Extensions

Extensions modul omogućava proširivanje funkcionalnosti Burp Suitea putem dodataka (ekstenzija). Korisnici mogu instalirati ekstenzije iz BApp Storea ili kreirati vlastite koristeći Burp Extender API. Ovo omogućava prilagođavanje alata specifičnim potrebama i dodavanje novih funkcija koje nisu dostupne u osnovnoj verziji. [4]

2.3. Usporedba alata *a priori*

U svojoj suštini ZAP i Burp suite Professional imaju iste funkcionalnosti. Kroz godine su OWASP i nezavisni programeri polako gradili alat koji sada već može parirati profesionalnom plaćenom alatu. Kao što se iz tablice 2.1. zaključiti ZAP po svemu parira Burp suite-u, no i dalje oba alata imaju prednosti i mana. Jedna od većih mana Burp Professional Suitea je njegova cijena gdje će nas samo jedna godina licence koštati 450 eura. Manjkavost ZAP-a je *Fuzzer*. U trenutku izrade rada funkcionalnost tog modula nije u potpunosti realizirana kao kod Burp-a što će kasnije biti pokazano. Funkcionalnosti koje ZAP ima a Burp suite nema su mogućnost automatizacije i HUD (engl. *Heads Up Display*). Automatizaciju ZAP sam prije spomenuo a HUD ću koristiti i objasniti prilikom testiranja. Što se tiče korisničkog sučelja oba alata pružaju lijepo i moderno sučelja, gdje Burp ima malu prednost zbog jasnoće i intuitivnosti kartica i modula.

Tablica 2.1. Burp suite modul i odgovarajuća alternativa realizirana u ZAP-u [5]

Burp Suite Professional značajka	ZAP alternativa
Collaborator	OAST Support Add-on
Comparer	Diff
Decoder	Encoder
DOM Invader	Eval Villian ekstenzija
Extender	Marketplace, Scripts
Intercept	Breakpoints
Intruder	Fuzzer*
Live scan	ATTACK Mode
Project Files	Session Files
Proxy	Proxy
Repeater	Manual Request Editor, Requester ekstenzija
Scanner	Active Scanner
Sequencer	Token Generation and Analysis
Target	Contexts

3. Testiranje

Jedini pravi način da se objektivno usporedi efektivnost oba alata jest da ih se testira na realnim i konkretnim situacijama. Za testiranje će se koristiti Burp Suite Web Security Academy laboratorijske vježbe. [6] Osim što postoje vodiči za Burp Suite postoje i vodiči za ZAP koje su korisnici pisali. [7, 8] Vježbe su odabrane uz pomoć OWASP Top 10 stranice [9] koja vodi brigu o broju i vrsti sigurnosnih rizika koji su do godine pisanja bili najčešći, te njihove povezane CWE-ove. Obraditi će se laboratorijske vježbe povezane sa sigurnosnim rizicima s vrha gore navedenog popisa. Specifično su izabrane one za koje postoje rješenja u ZAP vodiču ili čija se rješenja mogu lako pronaći.

3.1. Vidljiva pogreška bazirana na SQL injekciji

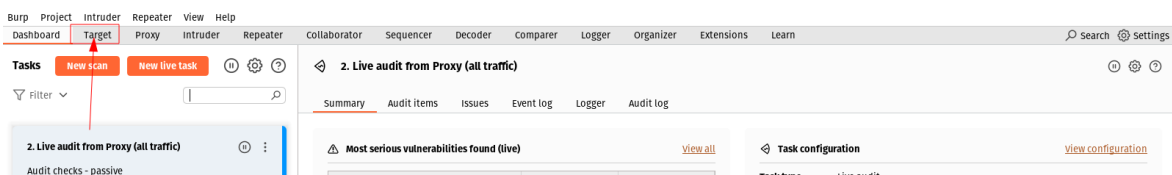
SQL injekcije su među najčešćim prijetnjama na Internetu. Činjenica koje to dokazuje je da su s ostalim vrstama injekcija 2017. predstavljale najrasprostranjeniji sigurnosni rizik ,a 2021. su bile čak na trećem mjestu po učestalosti. [9] SQL injekcija je tehnika napada u kojoj napadač ubacuje zlonamjerni SQL kod u polja za unos aplikacije kako bi izvršio neovlaštene radnje na bazi podataka. To može uključivati dobivanje osjetljivih podataka, izmjenu podataka, brisanje podataka ili izvođenje administrativnih operacija. SQL injekcija nastaje zbog neadekvatnog filtriranja ili validacije korisničkog unosa. [10, 11]

Laboratorijska vježba sadrži ranjivost na SQL injekcije koju će biti iskorištena. Od ostalih informacija znamo da aplikacija koristi 'kolačiće' za analitiku te provodi SQL upit s informacijama u podnesenim kolačićima. Rezultati SQL se ne vraćaju direktno već je potrebno izazvati grešku u ispisu. Baza podataka sadrži tablicu zvanu *users* sa stupcima *username* i *password*. Cilj laboratorijske vježbe je pronalazak načina da web aplikaciji 'iscuri' lozinka za korisnika *administrator* te se onda treba ulogirati s njihovim računom.

Ono što je također bitno za primjetiti je da se ne koriste podatci dobivene direktno iz malicioznog SQL upita, već grešku koja se zbog upita vraća. To uvelike otežava alatima da primjete takve ranjivosti kako *de facto* nema polja koje bi mogli iskoristiti da nam aplikacija da neke djelove baze podataka direktno. Umjesto toga cilj je iskoristiti lošu obradu grešaka. [12]

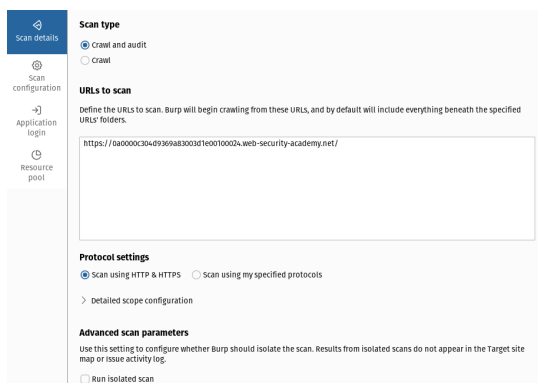
3.1.1. Vježba 1: Burp Suite

Nakon otvaranja alata potrebno je kliknuti na karticu *Target* kako je prikazano na slici 3.1.

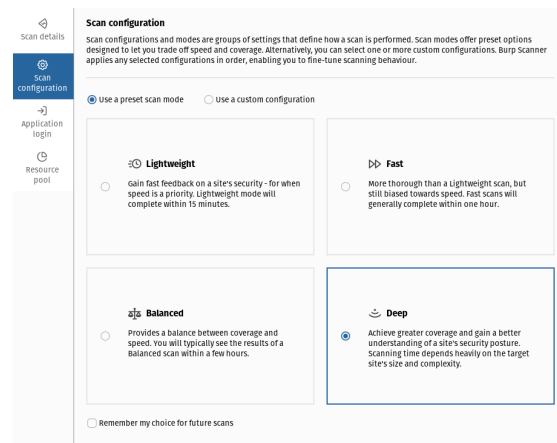


Slika 3.1. Popis svih kartica s naznačenom karticom *Target*

Ondje je prije svega potrebno uključiti proxy koji će biti korišten za presretanje zahtjeva i odgovora. Zatim je potrebno kliknuti na gumb *Open browser* te u adresnu traku zalijepiti adresu laboratorijske vježbe. Odmah prilikom otvaranja dedicanog pretraživača i unosom poveznice na vježbu Burp izgradi *sitemap* (mapu cijele stranice). Sada kako bi bilo moguće pronaći ranjivost koja će se iskoristiti za navedenu vježbu nužno je započeti skeniranje s kartice *Dashboard* > *New scan* > *Webapp scan*. Najprije se prikazuje pregled skena kao što je vidljivo na slici 3.2. Ovdje se mogu vidjeti generalne informacije o meti i trenutnim postavkama skena. Potom se konfigurira sken kao što je to vidljivo na slici 3.3. Prilikom ove i budućih vježbi koristiti će se *Deep* sken.



Slika 3.2. Pojediniosti skena



Slika 3.3. Konfiguriranje skena

Skeniranje traje <5 minuta te izvodi očekivane rezultate. Osim SQL ranjivosti, alat je pronašao nekolicinu informativnih detalja što se stranice tiče poput činjenice da komunikacija nije enkriptirana te kako nije definirana striktna sigurnosna politika za transport. Vezano za SQL injection alat je pronašao ranjivosti na čak 3 putanje */product*, */filter* i */my-account*. Za *product* i *filter* je koristio *payload* u kojem je u *trackingid* prvo dodao jednu ' a nakon što je aplikacija javila grešku dodao " na kraj *TrackingId*-a nakon koje je greška nestala. Zatim je pokušao napraviti novi payload viljiv na slici 3.4. nakon čega je primijetio da stranici treba dulje da odgovori te je deducirao da se najvjerojatnije radi o Postgres bazi podataka. Sa slike je vidljivo kako je iskorištena naredba za spavanje integrirana kao dio *TrackingId*-a koju baza nakon parsiranja izvršava.

Advisory	Request 1	Response 1	Request 2	Response 2	Request 3	Path to issue
Pretty	Raw	Hex				
1	GET /product?productId=3 HTTP/2					
2	Host: 0a5b003704f727ba833d4b8b002500ca.web-security-academy.net					
3	Accept-Encoding: gzip, deflate, br					
4	Accept: */*					
5	Accept-Language: en-US;q=0.9,en;q=0.8					
6	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.112 Safari/537.36					
7	Connection: close					
8	Cache-Control: max-age=0					
9	Referer: https://0a5b003704f727ba833d4b8b002500ca.web-security-academy.net/					
10	Cookie: TrackingId=MyQ9CdDsV8gLQkwZ'%7c%7cpg_sleep(20)-; session=raUnMJ2KSTISMvZovgiMG60xg60brDCc					
11						

Slika 3.4. Zahtjev koji je otkrio bazu podataka i iskoristio SQL ranjivost

Pod karticom *Advisory* s slike 3.4. je vidljivo koji CWE se krije iza otkrivene ranjivosti te detalje o ispitanoj ranjivosti. Sada je potrebno iskoristiti ovo znanje prilikom napada web aplikacije te potom izvršiti prijavu kao *administrator*. Prvo se navigira u karticu *Proxy* > *HTTP history*. Ondje se nalazi GET zahtjev za putanju */login* te je potrebno modificirati dio kolačića gdje piše *TrackingId* tako da na kraj vrijednosti stoji '. To će natjerati stranicu da izbací pogrešku iz kojeg se otkriva cijeli SQL upit što je vidljivo na slici 3.5.

```
Unterminated string literal started at position 52 in
SQL SELECT * FROM tracking WHERE id = '
JhlbaDuQVof0JeOw' '. Expected char
```

```
~~~~
```

Slika 3.5. Dobivena pogreška

Sada je potrebno modificirati taj zahtjev kako bi došli do lozinke korisnika *administrator*. Dobar bi pokušaj bio potpuno maknuti vrijednost *TrackingId* argumenta i zamijeniti ju sa zloćudnim SQL upitom. Koristeći Repeater modul (desni klik na zahtjev > *Send to Repeater*) te ćemo ondje izmijeniti vrijednost tako da sad u payload-u piše:

TrackingId=' AND 1=CAST((SELECT username FROM users LIMIT 1) AS int)–

Iz prijašnje greške 3.5. može se zaključiti da aplikacija traži samo jedan *TrackingId* te da je vjerojatno potrebno ograničiti upit na jedan redak baze kao i jedan stupac. Sada bi trebala doći nova greška koja kaže kako *administrator* nije tipa *integer*. Iz ove greške je očito da se *administrator* nalazi prvi na popisu. Kao što je vidljivo potrebno je pretvoriti polja koja nisu *integeri* u *int* upravo kako bi natjerali bazu da "procuri informacije". Ponovno ćemo se sada vratiti u *Repeater* te ponoviti prijašnji zahtjev s malom preinakom:

TrackingId=' AND 1=CAST((SELECT password FROM users LIMIT 1) AS int)–

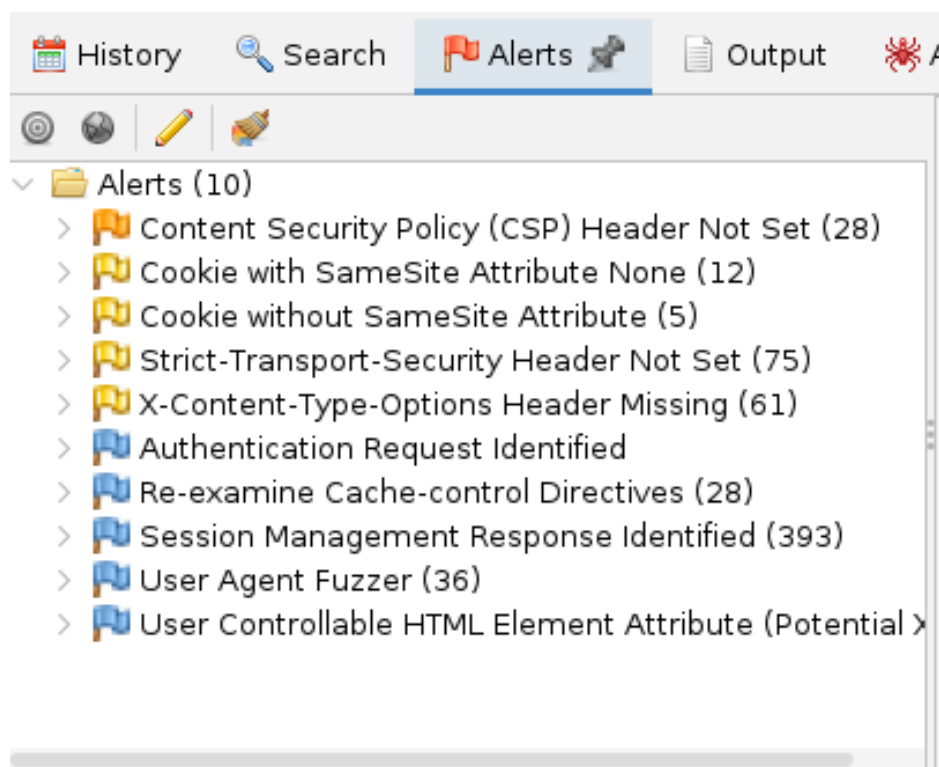
Ovaj upit napokon otkriva password za korisnika *administrator* 3.6. no kao grešku.

```
ERROR: invalid input syntax for type integer: "
vuapcoxchp374pzq5d31"
```

Slika 3.6. Pogreška koja ispisuje lozinku za administratora

3.1.2. Vježba 1: ZAP

Najprije u gornjem desnom kutu alat je potrebno postaviti *Mode* na *ATTACK*. Zatim je potrebno kliknuti na *Automated scan* i unijeti link mete. Skeniranje traje nešto dulje od Burpovog. Među alertima je vidljivo da ZAP nije primijetio da uopće postoji mogućnost SQL injection-a3.7. Prije je spomenuto kako ZAP ima Fuzzer ali nije trenutno funkcionalan. Razlog tomu je što ZAP trenutno nema mogućnost skeniranja HTTP zaglavlja što je veliki minus u odnosu na Burp. [5] Ovaj zadatak služi otkrivanju ove velike manjkavosti ZAP-a nad Burpom.



Slika 3.7. Manjak upozorenja na ranjivosti web aplikacije

Problem je moguće zaobići koristeći ZAP HUD. To je modul koji omogućava ručnu provjeru, izmjenu i slanje zahtjeva. Nakon što se uključi *Break* prilikom gledanja */login* putanje nakon klika na gumb Login. Ovdje je potrebno izmijeniti HTTP zaglavlje te modificirati kolačić kao što je prethodno to učinjeno za Burp. Naravno sada je otkriveno dobro rješenje na isti način kao i za Burp.

3.2. Dvofaktorska autentifikacija sa pogrešno konfiguriranom logikom

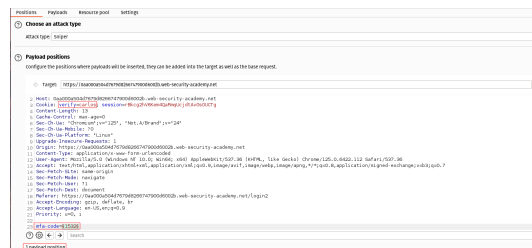
Dvofaktorska autentifikacija (2FA) je sigurnosni mehanizam koji koristi dva različita faktora za provjeru identiteta korisnika, kao što su lozinka (prvi faktor) i jednokratni kod poslan putem SMS-a ili aplikacije (drugi faktor). Iako 2FA značajno povećava sigurnost, može biti ranjiv na određene vrste napada ako nije pravilno implementiran. [13, 14] Primjeri loše implementacije su recikliranje sigurnosnih kodova, realizacija 2FA na klijentskoj strani, direktan pristup API-ju, loša programska potpora i mnogi drugi.

Ova laboratorijska vježba ima ranjivost u dvofaktorskoj autentifikaciji zbog pogrešne logike, konkretno uz zahtjev za unos sigurnosnog koda u kolačiću prenosi i ime korisnika te nema limit na slanje zahtjeva što uvelike kompromitira aplikaciju. Za rješavanje zadataka, potrebno je pristupiti računu korisnika *Carlota*. Kao informacije za vježbu su pruženi podaci za pristup korisniku *wiener* s lozinkom *peter* te pristup mail serveru. [15] Također je dan hint kako se korisnik *carlos* neće pokušati ulogirati za vrijeme trajanja napada.

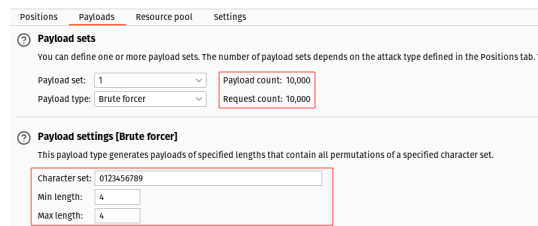
3.2.1. Testiranje: Burp Suite

Skeniranje se pokreće kao u prethodnoj vježbi. Kao u stvarnim situacijama, zanimljivi su POST zahtjevi koje skener šalje. Pokušat će se iskoristiti */login* putanja za rješavanje vježbe. Prije svega, pokušava se ulogirati s danim vjerodajnicama, nakon čega se traži unos 4-znamenkastog koda koji je poslan na mail. Već sada se uočava manjkavost realizacije. Ako kod ima samo 4 znamenke, to znači da postoji samo 10,000 mogućih kombinacija što nije puno za današnje sustave. Nakon što se uspije, pretražit će se povijest skeniranja da se vidi kako se zahtjev može izmijeniti.

Zahtjev sa putanjom */login2* šalje se na *Intruder* kao što je vidljivo na slici 3.8. Jednom kada se uđe u *Intruder*, vrijednost *verify* iz *wiener* mijenja se u *carlos* za GET zahtjev, što generira privremeni kod. Zatim se POST zahtjev iste putanje šalje na *Intruder* te se bira način rada *Sniper* i označava vrijednost polja *mfa-code* za modificirani payload što je vidljivo na ?? Burp će sada slati zahtjeve, svaki put modificirajući payload, kako bi pogodio točan kod za *carlosa*.



Slika 3.8. Zahtjev koji ćemo koristiti u Intruderu



Slika 3.9. Konfiguriranje payloada

Sada se traži odgovor koji se razlikuje od drugih ili po duljini ili po vrsti odgovora. U ovom slučaju razlikovat će se i po vrsti i po duljini odgovora. Jedan jedini payload vratio se s odgovorom 302, što označava redirekciju. Kada se pogleda vrijednost payloada za taj zahtjev, vidi se da je Carlosov trenutni kod 0100.

9. Intruder attack of https://0aa00a504d7679d8266747900d6002b.web-security-academy.net

Attack Save

Results Positions Payloads Resource pool Settings

Intruder attack results filter: Showing all items

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
11	0100	302	186			188	
0		200	247			3275	
1	0000	200	215			3275	
2	1000	200	208			3275	
3	2000	200	210			3275	
4	3000	200	220			3275	

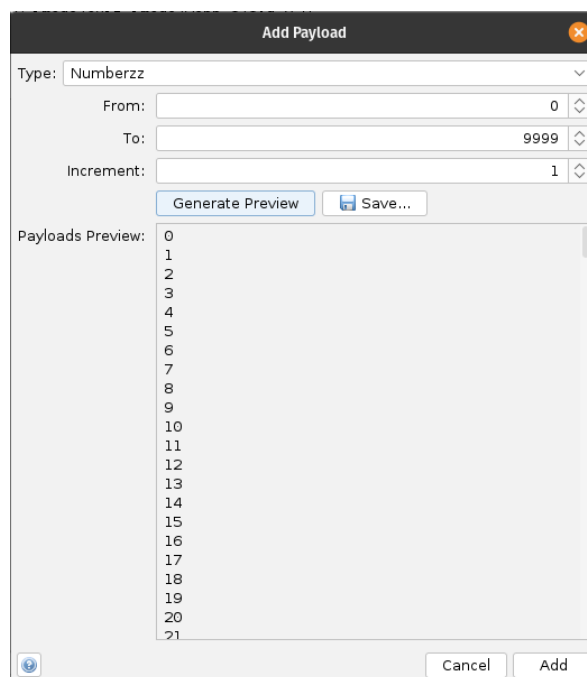
Slika 3.10. Prikaz napada uz modul Intruder

Odgovor se učita u pretraživač ili se kod ručno unese, čime se završava vježba. Pritom se mora paziti da se na preostalim zahtjevima za vrijednost *verify* upiše *carlos*.

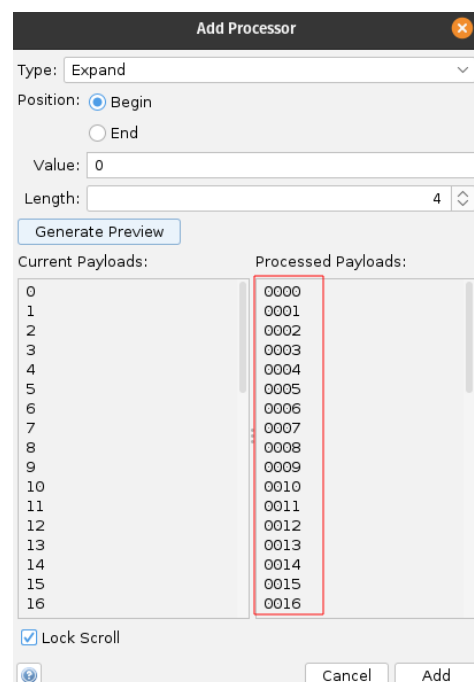
3.2.2. Testiranje: ZAP

Kako je već mnogo poznato o web aplikaciji, testiranje se može brže provesti. Sa ZAP HUD-om se ponovno normalno ulogira kao što je to učinjeno kod Burpa. Nakon što se ulogira s danim vjerodajnicama, otići će se u karticu *History* u glavnom izborniku. Ondje se izabire GET zahtjev za */login2* putanju, te mu se modificira vrijednost *verify* i zahtjev se šalje.

Zatim se pronade stari POST zahtjev za */login2*. Kako generator regularnih izraza nije funkcionalan u ZAP-u, potrebno je napraviti kratki "obilazak". Prvo se dodaje generator zahtjeva tipa *Numberzz*, koji stvara brojeve od 0 do 9999 što je vidljivo na slici 3.11. Nakon toga je iz slike 3.12. vidljivo da je potrebno namjestiti procesor za zahtjeve koji će zahtjeve pretvoriti u oblik koji je zapravo potreban.



Slika 3.11. Dodavanje generatora



Slika 3.12. Konfiguriranje procesora zahtjeva

Kao i u radu s Burpom sada se pronalazi odgovor koji se razlikuje te se šalje pregledniku. Nakon toga bi se trebalo pojaviti na zaslonu nešto kao na slici 3.13.

My Account

Your username is: carlos

Your email is: carlos@carlos-montoya.net

Slika 3.13. Prikaz uspješne prijave

3.3. Napad silom na lozinku koristeći lošu implementaciju promjene lozinke

Napad grubom silom na lozinku podrazumijeva isprobavanje svih mogućih kombinacija dok se ne pronađe točna. Kada se kombinira s loše implementiranom funkcionalnošću promjene lozinke, mogu se pojaviti ozbiljni sigurnosni problemi. Primjerice, ako sustav dopušta korisniku da promijeni lozinku bez provjere identiteta putem adekvatnih metoda (npr. slanje privremene lozinke na unaprijed spremljeni mail, korištenje dvofaktorske autentifikacije), otvara se prostor za iskorištavanje ranjivosti. U takvim scenarijima, napadač koji dobije pristup korisničkom računu putem metoda poput napada grubom silom ili socijalnog inženjeringa može lako promijeniti lozinku i time zaključati legitimnog korisnika. To se događa jer sustav ne provjerava dovoljno identitet korisnika prije nego što dopusti promjenu lozinke.

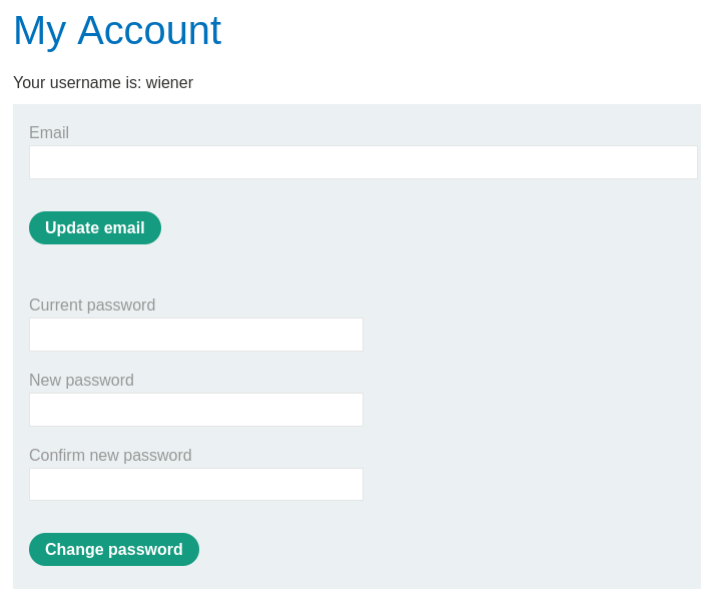
Kako bi se smanjili rizici, ključno je implementirati robustne sigurnosne mjere poput dvofaktorske autentifikacije, obavijesti o promjeni lozinke na registrirani virtualni sandučić (engl. *mail*), specifične fraze koje se generiraju za svakog korisnika pri registraciji i slična rješenja. Nažalost, napade grubom silom je gotovo nemoguće spriječiti te bi dizajneri trebali ciljati na to da je ovo najbolji mogući napad koji se pruža napadačima jer je ekstenzivan te u puno slučajeva zahtjeva puno vremena i resursa. [16]

Sigurnosni propust u funkcionalnosti promjene lozinke u ovoj laboratorijskoj vježbi čini ju podložnom napadima grubom silom. Za rješavanje ove vježbe, potrebno je iskoristiti listu potencijalnih lozinki kako bi se pristupilo Carlosovom korisničkom računu. Informacije koje su na raspolaganju uključuju vlastite vjerodajnice (`wiener:peter`), korisničko ime *carlos*, te listu kandidata za lozinku u obliku teksta gdje je svaka potencijalna lozinka zapisana u zaseban redak. [17]

3.3.1. Testiranje: ZAP

Prije nego što se započne s ručnim testiranjem aplikacije, provest će se automatizirani sken kako bi se dobila mapa povezanih stranica (engl. *sitemap*). Iako neće puno pomoći u rješavanju ove konkretne vježbe, uvijek je dobra praksa ispitati površinu napada kako bi se mogao pronaći dobar vektor napada. Sken sam po sebi ne otkriva previše, tako da će se pristupiti ručnom testiranju aplikacije uz pomoć ZAP HUD-a.

Prva stvar koja će se napraviti je prijava s danim vjerodajnicama kako bi se vidjelo kako aplikacija gradi zahtjeve za prijavu. Za to će poslužiti modul *Break*, koji omogućuje pregled i uređivanje svakog zahtjeva prije nego što ga preglednik pošalje i odgovora prije nego što ga preglednik primi. Iako naslov vježbe sugerira da vjerojatno neće biti moguće iskoristiti funkcionalnost prijave, dobro je biti temeljit i vidjeti postoji li neka ranjivost koja nije namjerno ostavljena. U ovom konkretnom slučaju, prijava je dobro implementirana, te će se nastaviti s testiranjem. Sada je zaslon nakon prijave nešto drukčiji nego na prijašnjim vježbama kao što je vidljivo na slici 3.14. Vidljivo je da je osim unosa maila moguća i promjena lozinke ukoliko je poznata trenutna.

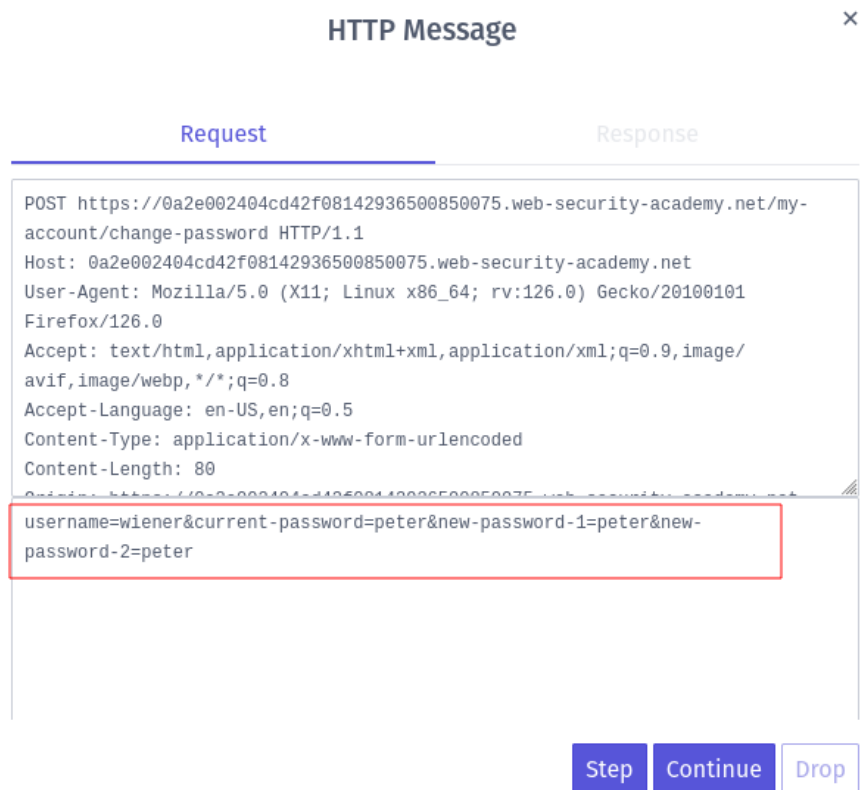


The screenshot shows a web interface titled "My Account" in blue text. Below the title, it says "Your username is: wiener". There are two main sections: "Email" and "Change password". The "Email" section has a text input field, a green "Update email" button, and a "Cancel" link. The "Change password" section has three text input fields labeled "Current password", "New password", and "Confirm new password", followed by a green "Change password" button and a "Cancel" link.

Slika 3.14. Sučelje nakon prijave sa dostupnim vjerodajnicama

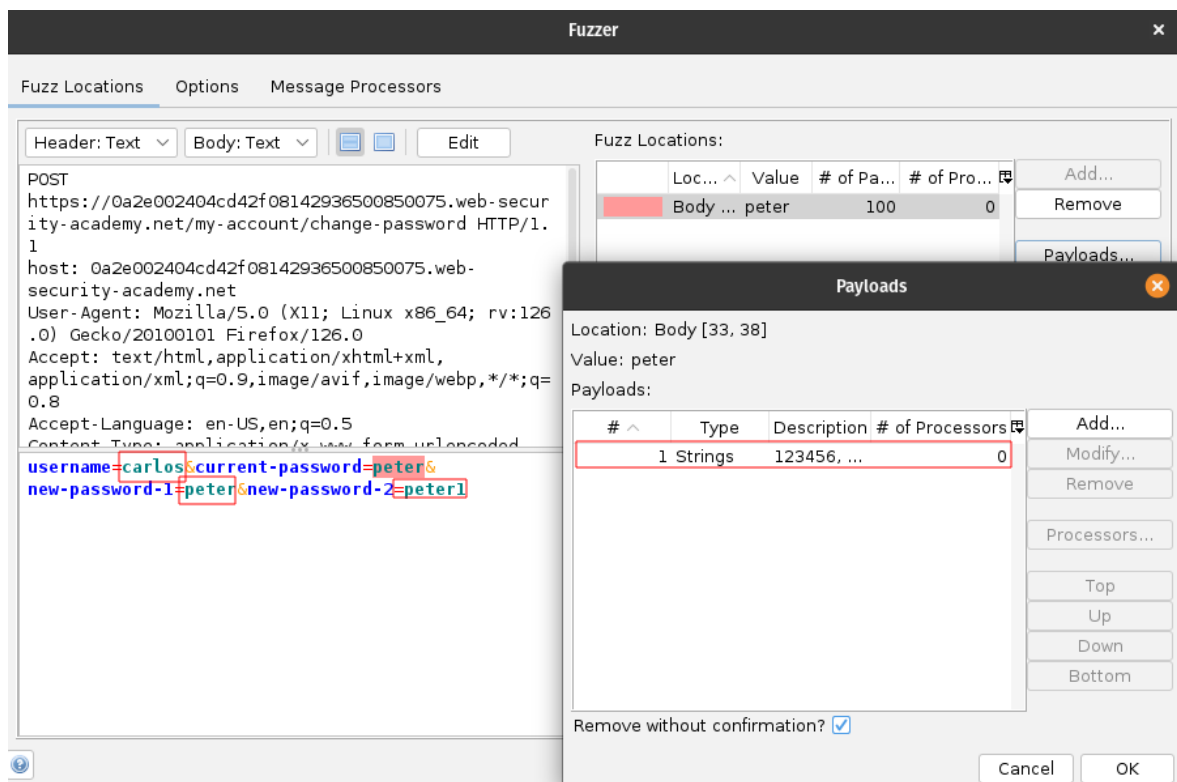
Očito je da se mora dokučiti kako iskoristiti opcije za mijenjanje lozinke. Pokušat će se promijeniti lozinka računu kojem se ima pristup kako bi se vidjelo kako web aplikacija gradi zahtjev. Ponovno će se koristiti *Break* funkcionalnost kako bi se presreo i uredio zahtjev

vidljiv na slici 3.15. Označeno je tijelo zahtjeva koje je potrebno urediti.



Slika 3.15. Zahtjev koji aplikacija generira

Iz slike 3.15. je vidljivo da se, kao u prijašnjim vježbama, može mijenjati korisničko ime kao i pripadajuća lozinka. Sada, kada su te putanje spremljene, koristit će se *Fuzzer* kako bi se razotkrila lozinka za korisnika *carlos*. Kako izmjena lozinki zapravo nije moguća, potrebno je na drugi način dokučiti koja je lozinka točna. Relevantni POST zahtjev šalje se u *Fuzzer*. Unutar *Fuzzera* korisničko ime *wiener* zamjenjuje se s *carlos*, a trenutna lozinka označava se kao mjesto na kojem će alat pogađati lozinke. Također, alat se opskrbljuje kandidatima za lozinku koji su dobiveni na početku vježbe, te se postavljaju različite nove lozinke. To se radi kako bi se umjesto pogreške da je lozinka netočna dobio odgovor koji javlja da se nove lozinke ne poklapaju. Primjer tako definiranog zahtjeva možemo vidjeti na slici 3.16.



Slika 3.16. Ispravno konfiguriran *Fuzzer*

Sada se traži odgovor koji, među onima kojima su zaprimljeni ima različitu duljinu odgovora. Uz njega nam piše i koju je lozinku alat pokušao poslati. Sada je potrebno tu lozinku iskoristiti da za prijavu kao *carlos* i tako završava ova laboratorijska vježba.

3.3.2. Testiranje: Burp Suite

Kao i za ZAP, aplikacija će se prvo skenirati kako bi se utvrdila površina napada. Kao i kod ZAP-a, skener nije utvrdio nikakve probleme. Unese se ispravna trenutna lozinka i dvije nove lozinke koje se ne podudaraju. Ovaj zahtjev POST /my-account/change-password šalje se u Burp Intruderu. Zbog prijašnjeg testiranja, zna se da je potrebno pratiti razlike u odgovorima. U Burp Intruderu parametar username mijenja se u carlos i dodaje se položaj za promjenu za parametar trenutne lozinke. Provjerava se da su parametri za nove lozinke postavljeni na dvije različite vrijednosti. Na kartici *Payloads* unosi se lista lozinki. Kada je napad završen, može se primijetiti da je jedan odgovor drukčije duljine i sadrži poruku "Nove lozinke se ne podudaraju". Lozinka koja je iskorištena u tom zahtjevu je lozinka za račun *carlos* što je vidljivo na slici 3.17.

The screenshot shows the Burp Suite interface during an intruder attack. The top bar indicates the target URL: `https://0a6000850487c34c82d69c82002a00e9.web-security-academy.net`. The 'Results' tab is active, displaying a table of intruder attack results. The table has columns: Request, Payload, Status code, Response re..., Error, Timeout, Length, and Comment. The first row (Request 27) is highlighted, showing a payload of '1qaz2wsx' with a status code of 200 and a response length of 4010. Below the table, the 'Response' tab is selected, showing the rendered HTML response. The response includes a heading 'My Account' and a message 'New passwords do not match' in a red box, followed by 'Your username is: carlos'. Below this is a form with fields for 'Email', 'Current password', and 'New password', and a green 'Update email' button.

Request	Payload	Status code	Response re...	Error	Timeout	Length	Comment
27	1qaz2wsx	200	143			4010	
2	password	200	394			4013	
8	111111	200	383			4013	
5	123456789	200	382			4013	
4	qwerty	200	377			4013	
10	dragon	200	370			4013	
0		200	341			4013	
1	123456	200	339			4013	
3	12345678	200	339			4013	
11	123123	200	296			4013	

Request Response

Pretty Raw Hex Render

My Account

New passwords do not match
Your username is: carlos

Email

Update email

Current password

New password

Slika 3.17. Kraj napada u *Intruderu*

Rješenje u ovom slučaju je **1qaz2wsx** što je i vidljivo iz slike 3.17.

4. Usporedba i nadomještanje funkcionalnosti

Oba alata imaju slične funkcionalnosti i mogu postići iste rezultate s minimalno različitim pristupom. ZAP je očito namijenjen za male tvrtke i pojedince, dok je Burp Suite komercijalan alat namijenjen za velike tvrtke i korporacije. Iako i velike tvrtke mogu koristiti ZAP, njihovi sigurnosni specijalisti moraju proći više obuke.

Razlike u korištenju su primijećene prilikom korištenja oba alata. Kada se pojavi problem s Burpom ili je potrebna informacija o njihovim alatima, uvijek se može osloniti na dobro napisanu dokumentaciju. *PortSwigger* (tvrtka koja je izgradila Burp Suite) osigurava profesionalne priručnike i video upute za korištenje svojih alata.

Kada se pojavi problem sa ZAP-om, uvelike se oslanja na rješenja koja su već dokumentirali članovi zajednice. Iako ZAP ima svoje priručnike i dokumentaciju, oni nisu na razini jasnoće, pokrivenosti i razumljivosti kao Burpova dokumentacija. Velika prednost ZAP-a je njegov HUD, koji je bio izuzetno koristan prilikom testiranja. U prednost ZAP-u također idu i dodaci, tj. ekstenzije, kojih ima puno više nego za Burp. Te ekstenzije mogu se pisati u *Javi*, *Pythonu*, *Rubyu*, *JavaScriptu*, itd. Osim toga, ZAP je moguće uključiti u skripte te na taj način automatizirati testiranje.

Primjer takve skripte:

```

import time
from zapv2 import ZAPv2

def zap_scan(target_url, api_key='vas_api_kljuc', log='zap_rezultati.txt'):
    zap = ZAPv2(apikey=api_key)
    zap.urlopen(target_url)
    time.sleep(2)
    zap.spider.scan(target_url)
    while int(zap.spider.status('')) < 100:
        time.sleep(2)
    zap.ascan.scan(target_url)
    while int(zap.ascan.status('')) < 100:
        time.sleep(5)
    alerts = zap.core.alerts() # rjecnik alarma
    with open(log, 'w') as f:
        for alert in alerts:
            f.write(f"Alert: {alert['alert']}\nRisk: {alert['risk']}\n \
URL: {alert['url']}\n Description: {alert['description']}\n \
Solution: {alert['solution']}\nReference: {alert['reference']}\n")

if __name__ == "__main__":
    target_url = input("Unesite URL zrtve: ")
    zap_scan(target_url)

```

Ispis 3.18. Jednostavna skripta za automatizaciju skeniranja u Pythonu [18]

Jedino što je potrebno napraviti je instalirati ZAP i Python, postaviti api ključ za korištenje, pokrenuti ZAP u *daemon* načinu rada te onda sa Pythonovim menadžerom za pakete instalirati paket za korištenje ZAP daemona.

```

./zap.sh -daemon -config api.key=your_api_key
pip install python-owasp-zap-v2.4

```

5. Zaključak

Ovaj rad dokazuje da OWASP ZAP može poslužiti kao adekvatna zamjena za Burp Suite Professional u mnogim aspektima testiranja sigurnosti web aplikacija. Provedena testiranja na konkretnim laboratorijskim vježbama pokazala su da oba alata mogu detektirati i iskoristiti iste ranjivosti, iako ponekad različitim pristupima. ZAP se istaknuo svojim intuitivnim sučeljem, posebice HUD-om koji olakšava interaktivno testiranje. Također, njegova otvorenost za proširenja i mogućnost automatizacije putem skripti čine ga izuzetno fleksibilnim alatom. S druge strane, Burp Suite Professional nudi nešto sofisticiraniji skup alata i bolje dokumentiranu podršku, što ga čini preferiranim izborom za veće organizacije i profesionalne penetracijske testere.

Ipak, postoje određene funkcionalnosti u kojima ZAP zaostaje, poput nemogućnosti skeniranja HTTP headera, što je bilo vidljivo u vježbi s SQL injekcijom. No, takvi nedostaci često se mogu nadomjestiti korištenjem dodataka ili alternativnih metoda. Ključna prednost ZAP-a je svakako njegova dostupnost kao besplatnog alata otvorenog koda, što ga čini pristupačnim pojedincima i manjim timovima koji si ne mogu priuštiti skupe licence. Uz to, aktivna zajednica koja stoji iza ZAP-a kontinuirano radi na poboljšanjima i proširenjima, čime se ovaj alat neprestano unapređuje.

Zaključno, iako Burp Suite Professional i dalje drži vodeću poziciju na tržištu, OWASP ZAP se pokazao kao snažan i sposoban alat koji može zadovoljiti većinu potreba u području testiranja sigurnosti web aplikacija. Izbor između ova dva alata često će ovisiti o specifičnim potrebama projekta, raspoloživom budžetu i stručnosti tima. U svakom slučaju, postojanje kvalitetne besplatne alternative poput ZAP-a značajno decentralizira tržište alatima za kibernetičku sigurnost, što u konačnici doprinosi sigurnijem web okruženju te osigurava nadmetanje i konstantnu inovaciju.

6. Literatura

- [1] OWASP, OWASP ZAP documentation, 2024., pristupljeno: 10.2.2024. [Mrežno]. Adresa: <https://www.zaproxy.org/docs/>
- [2] I. Homola, OWASP Zap: 8 Core Features (Pros & Cons), 2023-2-19., pristupljeno: 1.6.2024. [Mrežno]. Adresa: <https://www.codiga.io/blog/owasp-zap/>
- [3] G. Lyon, Nmap documentation, 2013., pristupljeno: 10.1.2024. [Mrežno]. Adresa: <https://nmap.org/docs.html>
- [4] PortSwigger, Burp Suite Professional - Features, 2024., pristupljeno: 5.5.2024. [Mrežno]. Adresa: <https://portswigger.net/burp/pro/features>
- [5] OWASP, Burp to ZAP Feature Map, 2024., pristupljeno: 10.6.2024. [Mrežno]. Adresa: <https://www.zaproxy.org/docs/burp-to-zap-feature-map/>
- [6] PortSwigger, Burp Suite Lab exercises, pristupljeno: 10.6.2024. [Mrežno]. Adresa: <https://portswigger.net/web-security/all-labs>
- [7] OWASP, PortSwigger Labs: 2FA Broken Logic, 6. travnja 2022., pristupljeno 10.6.2024. [Mrežno]. Adresa: <https://www.zaproxy.org/blog/2022-04-06-portswigger-lab-2fa-broken-logic/>
- [8] —, PortSwigger Labs: Password Brute-force via Password Change with ZAP, 6. travnja 2022., pristupljeno 10.6.2024. [Mrežno]. Adresa: <https://www.zaproxy.org/blog/2022-03-29-portswigger-lab-brute-force-password-change/>
- [9] —, OWASP, Top 10 Web Application Security Risks, 2021., pristupljeno: 7.6.2024. [Mrežno]. Adresa: <https://owasp.org/www-project-top-ten/>

- [10] J. Clarke, *SQL Injection Attacks and Defense*, C. Katsaropolous, Ur. Elsevier, 2012.
- [11] W3Schools, SQL Injection, pristupljeno: 10.6.2024. [Mrežno]. Adresa: https://www.w3schools.com/sql/sql_injection.asp
- [12] PortSwigger, PortSwigger, Lab: Visible error-based SQL injection, pristupljeno: 10.6.2024. [Mrežno]. Adresa: <https://portswigger.net/web-security/sql-injection/blind/lab-sql-injection-visible-error-based>
- [13] A. Dmitrienko, C. Liebchen, C. Rossow, i A.-R. Sadeghi, “On the (in) security of mobile two-factor authentication”, u *Financial Cryptography and Data Security: 18th International Conference, FC 2014, Christ Church, Barbados, March 3-7, 2014, Revised Selected Papers 18*. Springer, 2014., str. 365–383.
- [14] F. Aloul, S. Zahidi, i W. El-Hajj, “Two factor authentication using mobile phones”, u *2009 IEEE/ACS international conference on computer systems and applications*. IEEE, 2009., str. 641–644.
- [15] PortSwigger, PortSwigger, Lab: 2FA broken logic, pristupljeno: 10.6.2024. [Mrežno]. Adresa: <https://portswigger.net/web-security/authentication/multi-factor/lab-2fa-broken-logic>
- [16] L. R. Knudsen, M. J. Robshaw, L. R. Knudsen, i M. J. Robshaw, “Brute force attacks”, *The Block Cipher Companion*, str. 95–108, 2011.
- [17] PortSwigger, PortSwigger, Lab: Password brute-force via password change, pristupljeno: 10.6.2024. [Mrežno]. Adresa: <https://portswigger.net/web-security/authentication/other-mechanisms/lab-password-brute-force-via-password-change>
- [18] Y. Tarankov, OWASP ZAP API Python Script for Active Automated Scanning of Web Apps in Python 3, 21. rujna 2023., pristupljeno 13.6.2024. [Mrežno]. Adresa: <https://www.elinext.com/blog/owasp-zap-api-python-script-for-active-automated-scanning-of-web-apps-in-python-three/>

Upotreba alata ZAP kao zamjena za alat Burp Suite Professional

Ante Čavar

Sažetak

Cilj ovog rada bio je usporediti Burp Suite Professional i OWASP ZAP. Prije svega, uspoređene su njihove značajke po dokumentaciji, a zatim su testirani u sigurnom okruženju. Oba alata su se pokazala izuzetno korisnima i sveobuhvatnima, te je bilo koji problem moguće riješiti neovisno o izboru između ta dva alata. Uz to što su oba alata izuzetno korisna, također vrijedi nadodati kako oba posjeduju mogućnost proširenja kako bi zadovoljili različite potrebe. Pokazano je zašto je ipak ZAP malo bolji izbor, posebno za male tvrtke i pojedince, te kako ga je moguće automatizirati jednostavnim skriptama.

Na kraju je zaključeno kako su oba alata jednako bitna, te odluka koji će biti korišten u kojem trenutku ovisi o budžetu, znanju, iskustvu i situaciji. Ne postoji jedan alat koji može zamijeniti sve ostale, ali činjenica da postoji više od jednog je zdrava za razvoj, održavanje i unapređivanje svih alata na tržištu.

Ključne riječi: web sigurnost, OWASP, Burp Suite, penetracijsko testiranje, automatizacija, PortSwigger, ZAP

Using ZAP tool as alternative to Burp Suite Professional

Ante Čavar

Abstract

The aim of this study was to compare Burp Suite Professional and OWASP ZAP. Initially, their features were compared based on documentation, followed by testing in a secure environment. Both tools proved to be extremely useful and comprehensive, and any issue can be resolved regardless of the choice between these two tools. In addition to being extremely useful, it is worth noting that both tools have the capability for extensions to meet various needs. It has been demonstrated why ZAP is a slightly better choice, especially for small companies and individuals, and how it can be automated with simple scripts.

In the end, it was concluded that both tools are equally important, and the decision on which tool to use in a given situation depends on the budget, knowledge, experience, and context. There is no single tool that can replace all others, but the existence of more than one tool is

beneficial for the development, maintenance, and improvement of all tools on the market.

Keywords: web security, OWASP, Burp Suite, penetration testing, automation, PortSwigger, ZAP

Skraćenice

SUT	<i>System Under Test</i>	sustav koji se testira
OWASP	<i>Open Web Application Security Project</i>	
HTTP	<i>Hypertext Transfer Protocol</i>	Protokol za prijenos Hypertext datoteka
HTTPS	<i>HTTP Secure</i>	sigurni HTTP
ZAP	<i>Zed Attack Proxy</i>	Zed napadački posrednik
HUD	<i>Heads Up Display</i>	Zaslona sa dodatnim informacijama
CWE	<i>Common Weakness Enumeration</i>	(enumerirane) najčešće ranjivosti
SQL	<i>Structured Query Language</i>	Strukturirani jezik zahtjeva
2FA	<i>Two Factor Authentication</i>	Dvofaktorska autentifikacija
API	<i>Application Programming Interface</i>	sučelje za programiranje aplikacija
MITM	<i>Man In The Middle</i>	tihi posrednik, čovjek u sredini tj. između dva uređaja ili sustava