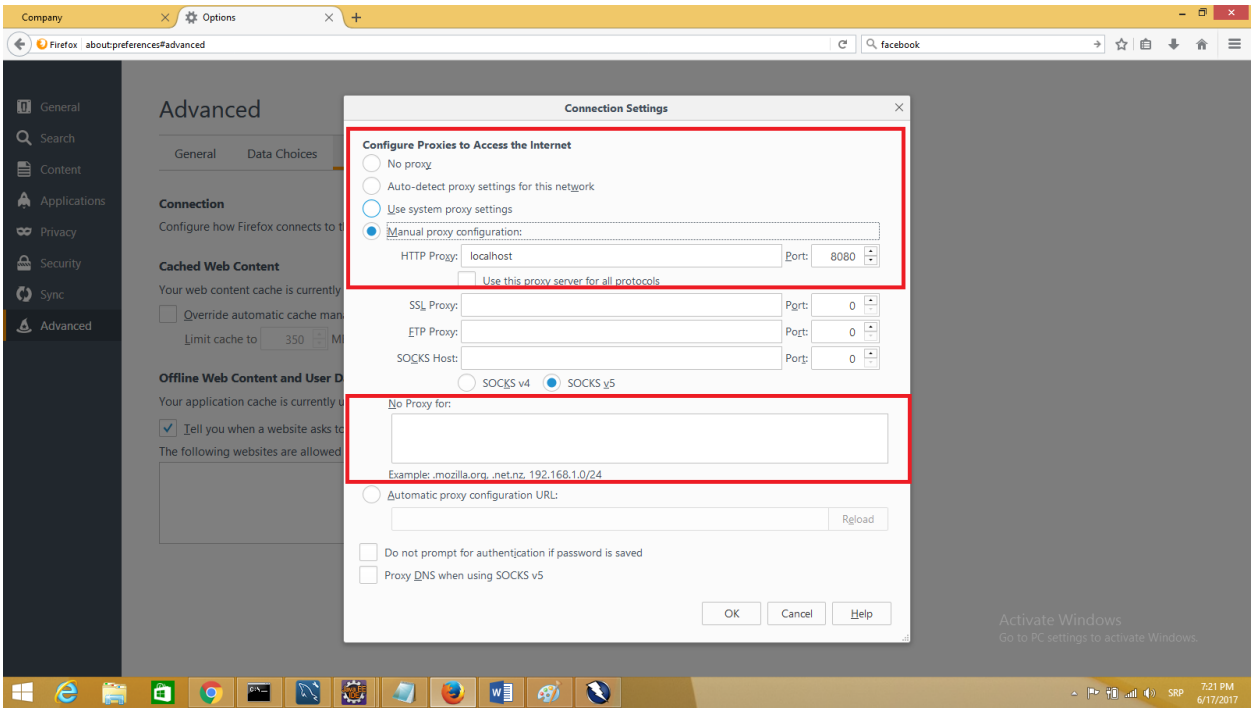
OWASP ZAP – Alat za penetraciono testiranje – Korisničko uputstvo

Da bismo omogućili da OWASP ZAP može da radi kao presretajući proksi (*intercepting proxy*), moramo to da podesimo u željenom pretraživaču. U Mozilli to radimo na sledeći način: *Options -> Advanced -> Network*, i kod pasusa "*Connection*" dugme "*Setings*".

Dobiće se dijalog kao na slici 1. Za „*Configure Proxies to Access the Internet*" označiti „*Manual proxy configuration*" i podesiti port na 8080. OWASP ZAP po inicijalnim podešavanjima osluškuje port 8080, ali to može da se promeni. Ukoliko želimo da nam aplikacija zauzme taj port, moramo u ZAP-ovim podešavanjima da promenimo podrazumevani port. (*Tools -> Options -> Local Proxy*)

Bitno je napomenuti da je potrebno da se *localhost* obriše iz "*No Proxy for*" panela.



Slika 1 - Podešavanja u Mozilli

Ovim smo omogućili da saobraćaj koji se odvija između pretraživača i aplikacije prolazi kroz ZAP, tako da može da presretne sve zahteve i odgovore, kao i da ih menja.

Spider i AJAX Spider

Spider je alat koji se koristi za automatsko pronalaženje url-ova na datom sajtu. Kreće od neke liste url-ova i identifikuju se svi hiperlinkovi, koji se potom dodaju na inicijalnu listu. Postupak se nastavlja sve dok se novi url-ovi pronalaze ili se ispuni neki od uslova za prekid, na primer, ukoliko je isteklo prethodno određeno vreme.

Ajax Spider je značajno sporiji, ali bolje radi na stranicama koje imaju dosta *Javascript* koda. Ako se testira aplikacija bazirana na AJAX-u, onda je Ajax Spider preporučeno rešenje. Dok radi, pokrene odvojeni pretraživač i "prolazi" kroz aplikaciju.

Za pronalaženje što više resursa (url-ova) najbolje je koristi manuelno pretraživanje, uz dodatak i Spider-a i AJAX Spider-a. Prednost kod manuelnog unosa je ta što korisnik unosi smislenije podatke u forme, dok su Spideri pogodni za pronalaženje skrivenih resursa. U kompleksnijim formama za unos može da se desi da ne uspe da unese validne podatke i pronađe neki resurs, na primer, kod broja računa, gde je potrebno uneti podatke u tačno određenom formatu.

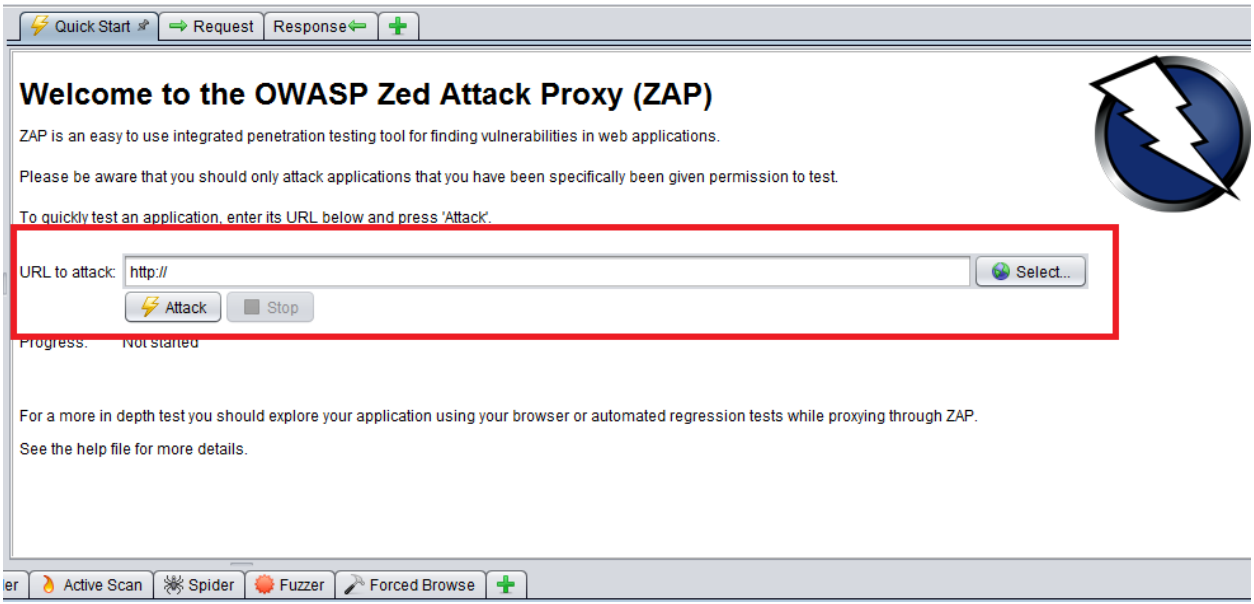
Active i Pasive scan

Pasivni Scaner radi sve vreme i samo proverava zahteve i odgovore. OWASP ZAP, dok god je pokrenut, će pratiti sav saobraćaj na stranicama koje posetimo. Na taj način, možemo da vidimo GET i POST zahteve i odgovore koji se razmenjuju, njihovo zaglavlje i strukturu.

Aktivni Scaner se koristi za pronalaženje slabosti u aplikaciji i izvodi širok dijapazon napada. Zbog toga ne smemo da ga koristimo ukoliko nemamo dozvolu od vlasnika. Moguće je pokrenuti napad na celu aplikaciju ili na neki konkretni url. Pre pokretanja se mogu označiti napadi koje želimo da se koriste, kojim intezitetom, odnosno koliko će ih količinski poslati, i još razna, naprednija, podešavanja.

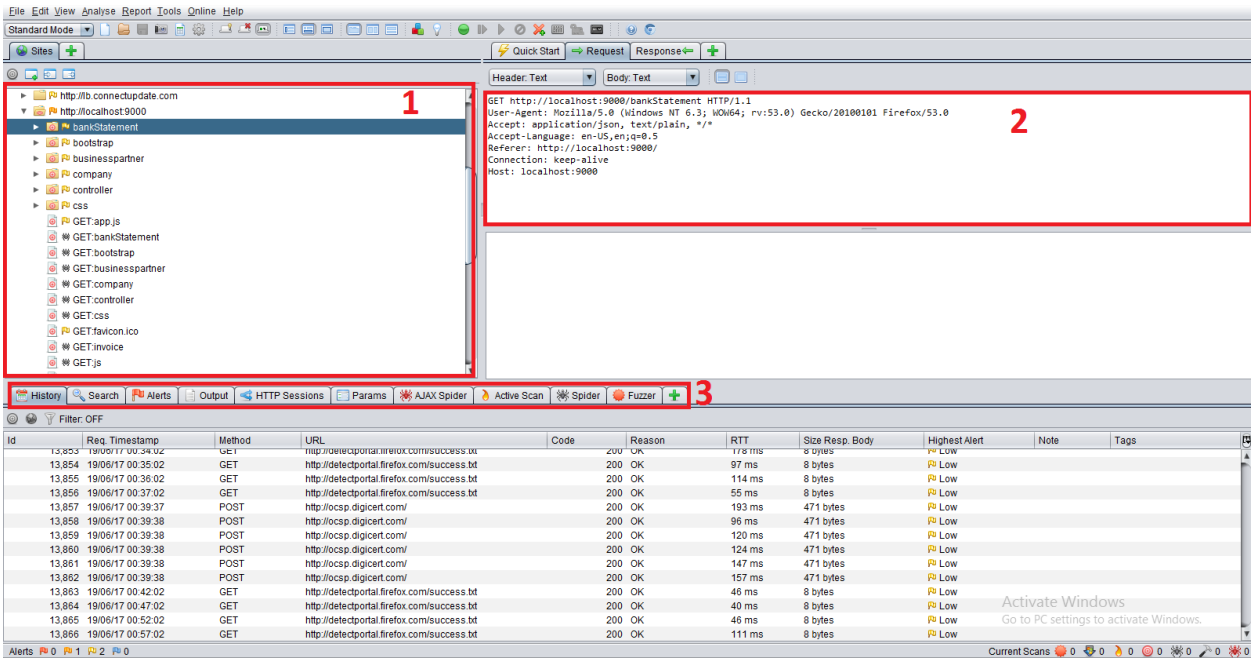
Quick start

Prvo, unesemo url aplikacije koju želimo da testiramo i zatim kliknemo „*Attack*“. Program pokreće pauka koji traži moguće url-ove. Po svom okončanju (o radu pauka će biti više u daljem tekstu), pokreće *Active scan* nad pronađenim url-ovima. Mana ovog pristupa je ta što samo pokretanje pauka neće pronaći sve moguće tačke za napad i što ne može da “zaobiđe” autentifikaciju, tako da će napad ostati na login formi.



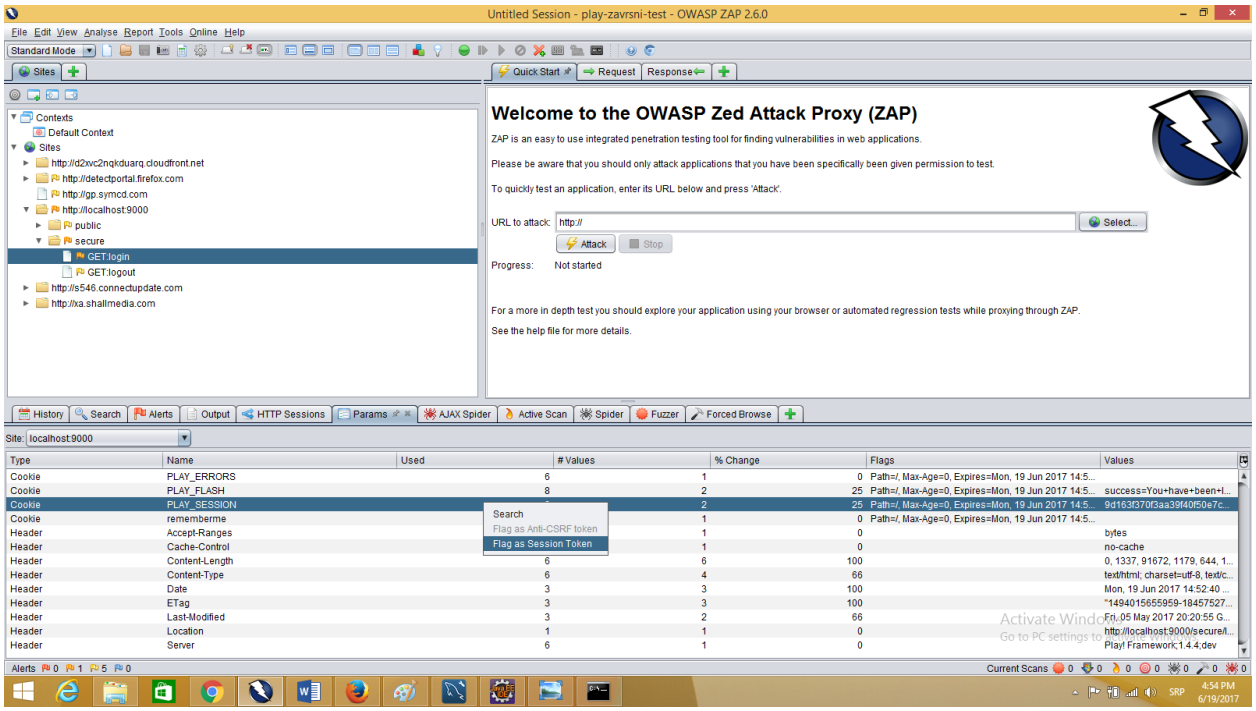
Slika 2 - Quick Start panel

AJAX spidering korak po korak



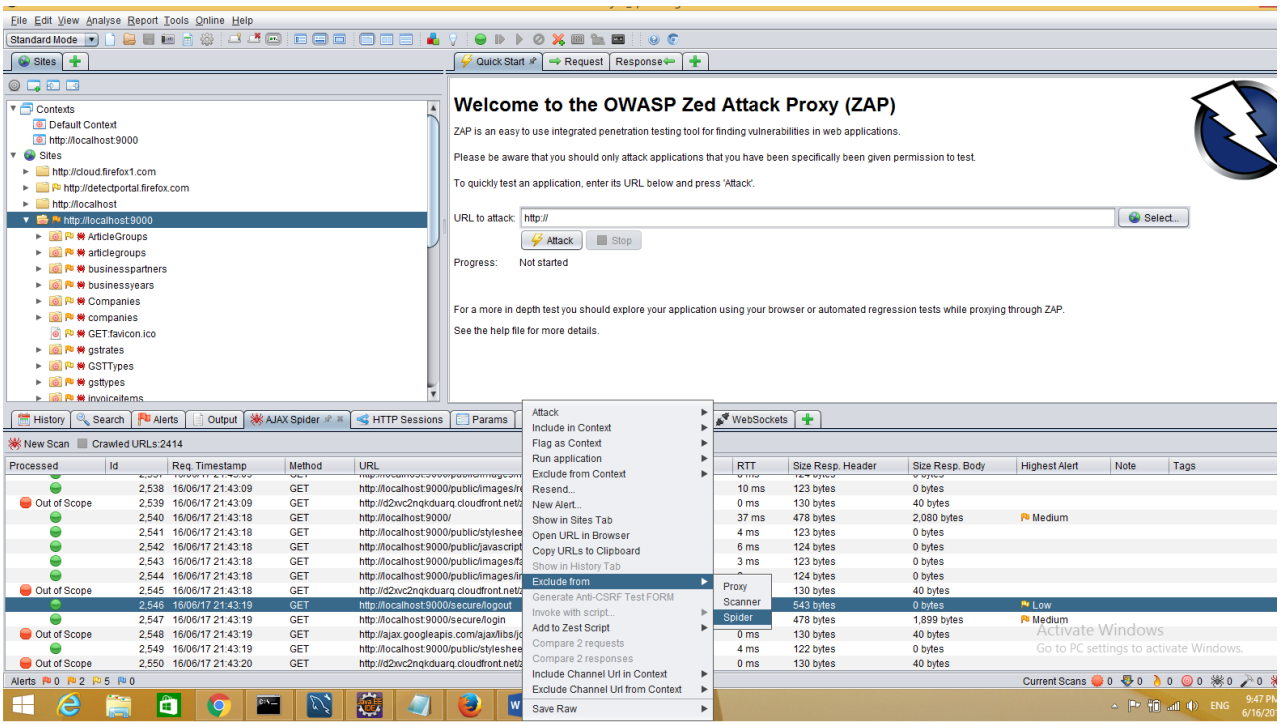
Slika 3 - prikaz GUI-a

1. Otvorimo web pretraživač - u ovom slučaju, Mozilla Firefox, jer smo u njemu ZAP podesili kao proxy
2. Odemo na sajt koji želimo da istražimo (npr. *localhost:9000*)
3. U "*History*" tabu OWASP Zap-a imamo listu GET i POST poziva. (History tab je prikazan na slici 3, među tabovima označenim rednim brojem 3)
   U gornjem levom panelu se nalazi istorija posećenih sajtova. (slika x, uokvireno i označeno sa 1)
4. U donjem, "*Params*" tabu, možemo da nadjemo sve parametre koji su se razmenjivali. Potrebno je da pronadjemo *session cookie* i da ga označimo, desni klik -> *Flag as session token*. (slika 4)



Slika 4 - Flag as Session Token

5. Nakon toga odemo u *HTTP Sessions* tab i kliknemo na "*New session*".
6. Na sajtu se logujemo. Ovaj korak je potreban da bi ZAP napravio sesiju sa *cookie*-jem.
7. Sledeće, dodamo sajt u *context*, desni klik -> *Include in context*. *Context* određuje skup url-ova koji će se razmatrati.
8. Desni klik -> *Attack*-> *AJAX spider*; Uključimo opciju "*in scope*".
9. U donjem desnom delu prozora ćemo videti da je AJAX Spider:1.
10. Napad je pokrenut, otvoriće se poseban pretraživač gde će AJAX Spider da istražuje.
11. Ako se izloguje, nadjemo akciju u AJAX Spideru i desni klik -> *exlude from* -> *Spider*. U novijim verzijama aplikacije će se sam opet ulogovati. (slika 5)



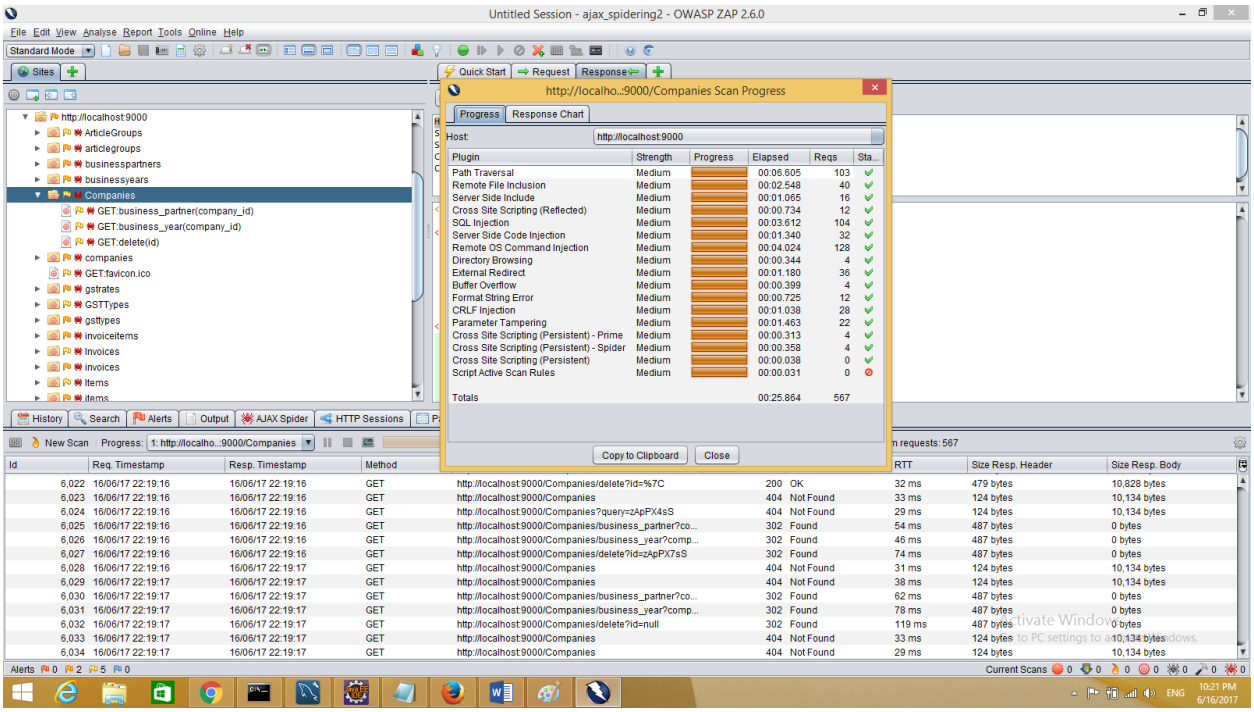
Slika 5 - Exclude from spider

Dodatno pronalaženje resursa

Kada AJAX Spider završi sa svojom pretragom, preporučljivo je pokrenuti i običnog Spider-a. Desni klik na url početne stranice -> *Attack* -> *Spider*. Za kraj je potrebno manuelno pretražiti sajt, pogotovo praviti smislene unose u forme. Sada imamo skup resursa za napad.

Pronalaženje propusta

Desni klik na url početne strane (http://localhost:9000 u ovom slučaju) -> *Attack* -> *Active scan* -> *Start scan*. (može se pokrenuti i na konkretnom url-u)



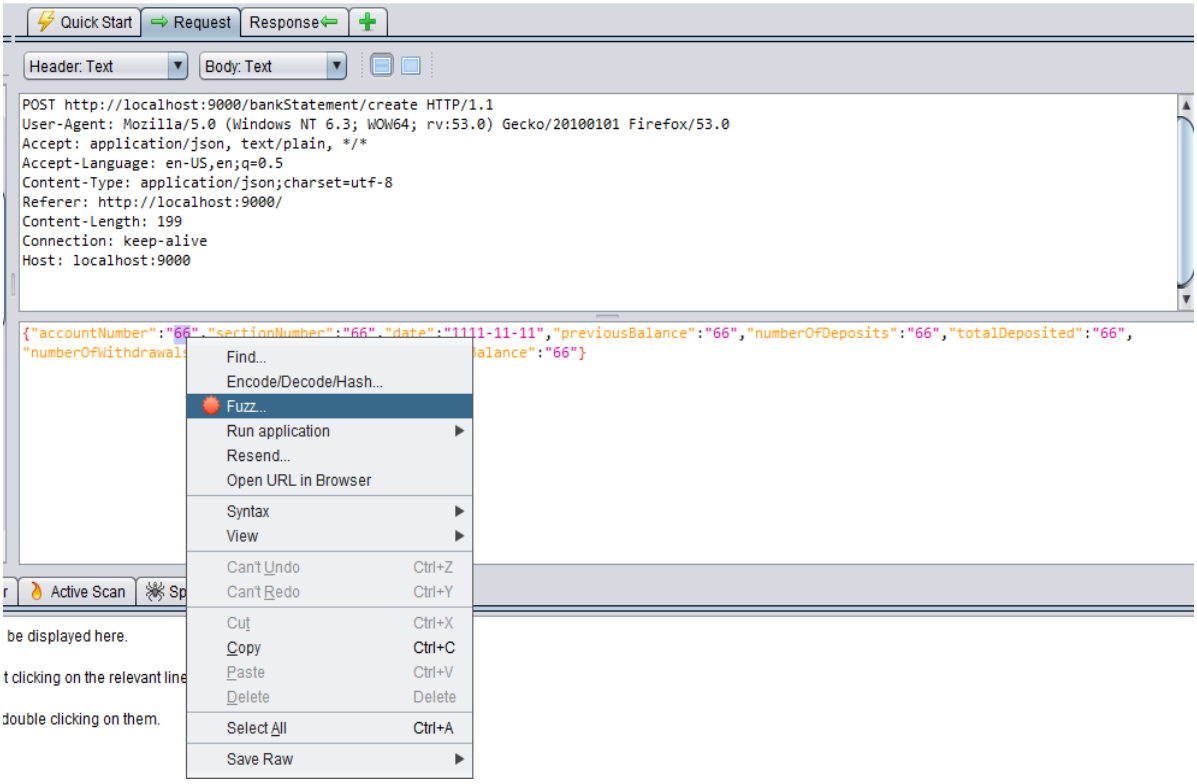
Slika 6 - Prikaz svih napada na jedan url

Pokrenuli smo napad, a po okončanju možemo da vidimo listu propusta u "*Alert*" tabu.

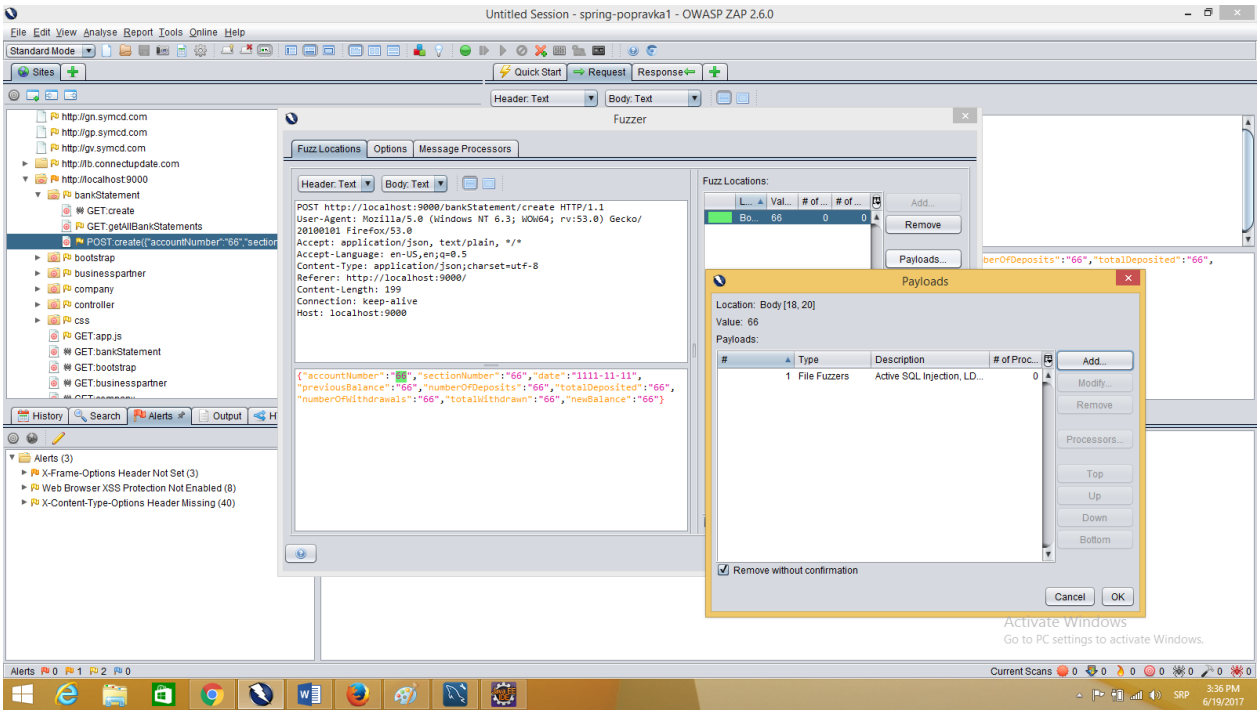
Fuzzer

Služi za “finije” napade. Označimo deo zahteva i podesimo listu podataka sa kojim ćemo da napadnemo.

Izaberemo neki poziv, i u *Request* delu označimo željeni deo. (slika 7). U *Payload*-su odaberemo kojim fajlom ćemo da napadamo, recimo specijalizovanim za *SQL Injection*. (slika 8)



Slika 7 – Fuzz napad



Slika 8 – Podešavanje Fuzz napada

Forced browse

Forced browse pokušava da nadje fajlove i direktorijume (koje ne bismo trebali da vidimo). Izabere se kojim fajlom će se vršiti napad. Fajl sadrži veliki broj imena direktorijuma i fajlova. ZAP pokušava direktno da im pristupi, a ne preko linkova.

Generisanje izveštaja

Izveštaj se može generisati klikom na „*Report*“ stavku menija, pa „*Generate HTML report*“.