HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Provjera ranjivosti web aplikacija korištenjem WebScarab alata

CCERT-PUBDOC-2007-07-199

CARNet CERT u suradnji s LS&S

Sigurnosni problemi u računalnim programima i operativnim sustavima područje je na kojem CARNet CERT kontinuirano radi.

Rezultat toga rada ovaj je dokument, koji je nastao suradnjom CARNet CERT-a i LS&S-a, a za koji se nadamo se da će Vam koristiti u poboljšanju sigurnosti Vašeg sustava.

CARNet CERT, www.cert.hr - nacionalno središte za **sigurnost računalnih mreža** i sustava.

LS&S, www.lss.hr - laboratorij za sustave i signale pri Zavodu za elektroničke sustave i obradbu informacija Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu.

Sadržaj

1. Uvod

Sve veća zastupljenost računala u svijetu i domovima pojedinaca dovela je do pojave migracije poslovanja iz tradicionalnih oblika vezanih uz papir prema elektroničkim oblicima mahom utemeljenim na Internet tehnologijama.

Izravna posljedica eksplozije oglašavanja i prodavanja preko Interneta je i sve veći broj tehnologija (npr. Java, Flash, Php, ASP...) koje omogućuju jednostavnu izradu aplikacija. Ovdje je potrebno primijetiti kako su pri prelasku na Internet poslovanje, osim prednosti, preuzeti i neki od problema računalnog svijeta. Računalna sigurnost je jedan od njh.

Nepoznavanje tehnologije koja se primjenjuje u poslovanju rezultira neočekivanim, izrazito neugodnim, problemima. Šteta do koje takav propust može dovesti je ogromna, a osim novčanih gubitaka, može uzrokovati i gubitak povjerenja klijenata, što na koncu može dovesti do propasti poslovanja.

Jedan od prvih koraka zaštite od spomenutog scenarija je upoznavanje prednosti i nedostataka korištenih tehnologija. Web tehnologije u ovoj priči predstavljaju sučelje prema vanjskom svijetu i zbog toga su kritična točka sigurnog poslovanja.

Jedan od koraka u procesu izrade svake pa i web aplikacije je njeno testiranje. Osim uobičajenog testiranja stabilnosti potrebno je primijeniti i provjere koje uključuju testiranje sigurnosti. Ovim se provjerama utvrđuje otpornost programskog koda ili tehnologije na propuste vezane uz njene osobine.

U daljnjem tekstu ovog dokumenta opisan je WebScarab - alat koji omogućuje testiranje sigurnosnih značajki web aplikacija. Osim osnovnih postavki, instalacije i uklanjanja, temeljito su obrazložene i neke napredne sigurnosne provjere. Dodavanjem nekih od njih u ciklus testiranja aplikacije povećava se razina njene , ali i sigurnosti cijelog sustava u kome se ona nalazi.

2. OWASP projekti

2.1. OWASP

OWASP (eng. *Open Web Application Security Project*) je organizacija koja djeluje s ciljem pronalaženja i rješavanja uzroka nesigurne programske potpore. Kako bi programerima i testerima olakšali posao izradili su nekolicinu projekata vezanih uz sigurnost web aplikacija. Jedan od njih je i WebScarab, alat namijenjen pronalaženju propusta. Osim njega tu je i *WebGoat* koje je orijentiran na edukaciju korisnika o teoretskoj i praktičnoj podlozi sigurnosnih propusta, te tehnikama njihovog izbjegavanja. Ostali projekti OWASP organizacije uključuju platforme za testiranje sigurnosti JavaScript aplikacija (CAL9000), koda Java aplikacija (LAPSE), XML i Ajax sučelja (*Interceptor*) i mnoge druge.
Neki dijelovi prethodno spomenutog *WebGoat* projekta korišteni su prilikom analize mogućnosti WebScarab alata, o čemu se više informacija može pronaći u ostatku dokumenta.

2.2. WebScarab

WebScarab je platforma koja pruža podlogu za izvođenje većeg broja različitih modula namijenjenih analizi sigurnosti aplikacija koje komuniciraju putem HTTP i HTTPS protokola. Napisan je u programskom jeziku Java te se kao takav može pokrenuti na svim operacijskim sustavima za koje postoji Java programski paket. Najčešće se koristi kao posredni poslužitelj (eng. *proxy*) što mu omogućuje presretanje HTTP/HTTPS prometa te izravnu izmjenu podataka u zahtjevima i odgovorima. WebScarab moduli među ostalim uključuju stavke iz slijedeće liste.

- *Fragments* – modul koji iz HTML odgovora izdvaja skripte i komentare. Izdvojeni podaci mogu se iskoristiti za pronalaženje skrivenih poveznica (eng. *link*) i za bolje razumijevanje rada stranice.
- *Proxy* – modul koji djeluje kao posredni HTTP poslužitelj. Predstavlja priključnu točku u kojoj se WebScarab sustav uključuje u konverzaciju između klijenta i web poslužitelja. Osim nezaštićenih HTTP konverzacija, modul se može uključiti i u HTTPS komunikaciju pri čemu dogovara posebnu SSL konekciju između sebe i preglednika. Na taj su mu način dostupni prenošeni podaci u svom izvornom, nezaštićenom, obliku.
- *Manual interception* – modul koji korisniku dopušta ručnu izmjenu presretnutih HTTP i HTTPS zahtjeva i odgovora.
- *BeanShell* – jezik nalik na Javu koji korisniku omogućuje pisanje skripti, pri čemu mu u zadavanju operacija nudi funkcionalnost Java okruženja.
- *Reveal hidden fields* – modul omogućava pretvaranje svih skrivenih polja u tekstualna polja, time ih čini vidljivima i omogućuje njihovu izmjenu.
- *Bandwith simulator* – korisniku omogućuje simuliranje sporije mreže te tako daje uvid u ponašanje stranice kada joj se pristupa,primjerice, s modemske veze.
- *Spider* – identificira nove URL (eng. *Uniform Resource Locator*) adrese, te ih pohranjuje i dohvaća na zahtjev.
- *Manual request* – omogućava izmjenu i ponovno slanje prijašnjih zahtjeva.
- *SessionID analysis* – prikuplja i prikazuje podatke vezane uz identifikaciju korisničke sjednice. Uočavanje eventualnih uzoraka i pravilnosti prepušta korisniku, jer je čovjek u tome daleko efikasniji.
- *Scripted* – modul koji korisniku omogućuje izvršavanje *BeanShell* i drugih skripti.
- *Parameter fuzzer* – obavlja automatiziranu zamjenu parametarskih vrijednosti te tako pokušava pronaći XSS (eng. *Cross Site Scripting*) i *SQL Injection* ranjivosti.
- *Search* – modul koji korisniku omogućuje zadavanje BeanShell izraza, čijim izvođenjem se identificiraju "zanimljive" konverzacije.
- *Compare* – modul koji uspoređuje više HTTP/HTTPS odgovora i utvrđuje stupanj njihove sličnosti.
- SOAP – modul koji prevodi WSDL (eng. *Web Service Definition Language*) i prikazuje različite opcije i potrebne parametre, te omogućuje njihovu izmjenu prije slanja poslužitelju.

- *Extensions* – modul koji automatski provjerava postoje li neke slučajno ostavljene datoteke u direktoriju poslužitelja (npr. privremene datoteke `.tmp`, `.bak` i td.). Korisnik može proizvoljno definirati koje će datoteke biti pregledavane.
- XSS/CRLF – modul koji pasivno traži korisnički zadane podatke u tijelu i zaglavlju HTTP odgovora te pokušava identificirati potencijalnu osjetljivost na napad dijeljenjem HTTP odgovora (eng. *HTTP Response Splitting*).

3. WebScarab

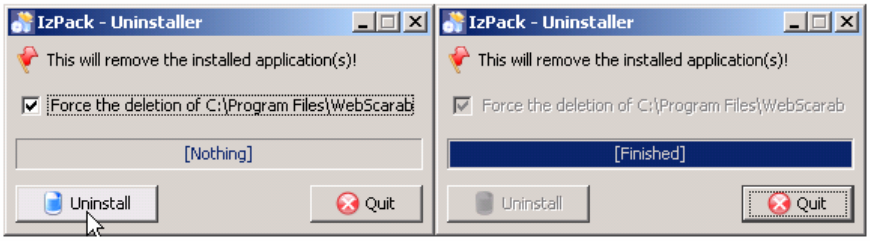
3.1. Instalacija

Preduvjet rada WebScarab alata je prisutnost JRE (eng. *Java Runtime Engine*) programskog paketa koga je moguće preuzeti sa stranice http://www.java.com/en/download/manual.jsp. Nakon provjere ispunjenosti ovog preduvjeta može se pokrenuti WebScarab instalacijska datoteka.
Pokretanjem se prikazuje poruka dobrodošlice te nekoliko osnovnih informacija o projektu. Postavke koje je moguće izmijeniti tijekom instalacije su:

- putanja do direktorija gdje će biti instalirane datoteke,
- mjesta na koja se žele postaviti kratice za pokretanje programa,
- postavka instalacije dodatnog sadržaja (dokumentacije i izvornog programskog koda) i
- postavka generiranja skripte koja će zapamtiti odabrane opcije, te ubrzati eventualnu ponovnu instalaciju programskog paketa.

3.2. Uklanjanje programskog paketa

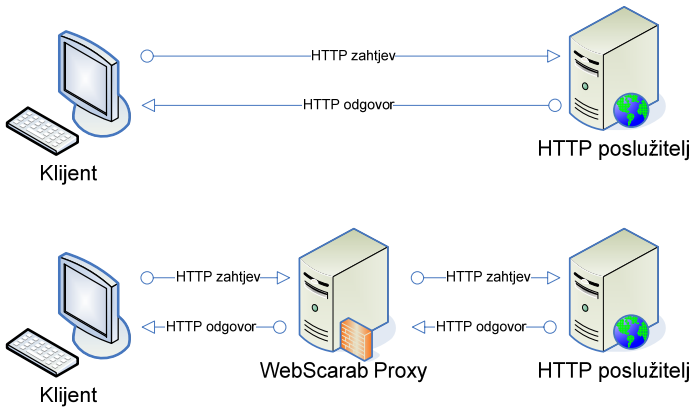
Kako bi uklonili WebScarab programski paket potrebno je u direktoriju koji je prilikom instalacije odabran kao odredište odabrati direktorij *Uninstall* i potom pokrenuti datoteku *uninstaller.jar.* Na slijedećoj slici je prikazan slijed dijaloga prilikom procesa uklanjanja.



Slika 1. Uklanjanje programskog paketa

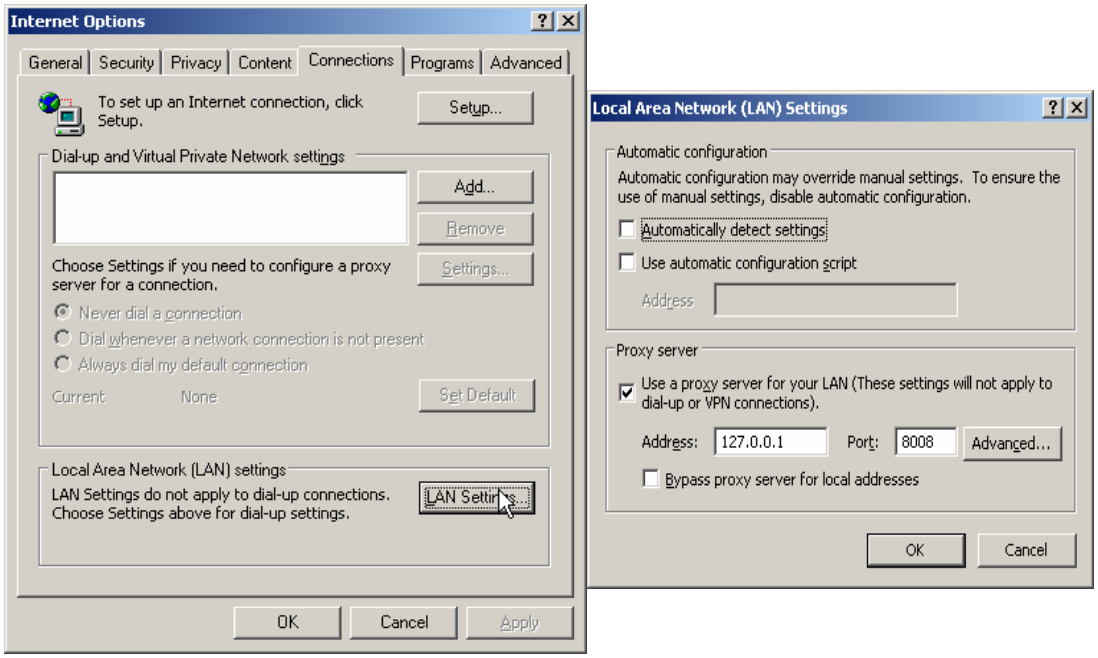
3.3. Korištenje WebScarab alata

Jedna od osnovnih mogućnosti WebScarab alata je presretanje HTTP i HTTPS komunikacije. Kako bi se ona iskoristila potrebno je pokrenuti WebScarab *Proxy* modul i izmijeniti postavke klijenta (najčešće web preglednika) tako da koristi posredni poslužitelj.
Na slijedećoj slici shematski je prikazano priključivanje WebScarab sustava u uobičajenu HTTP komunikaciju.



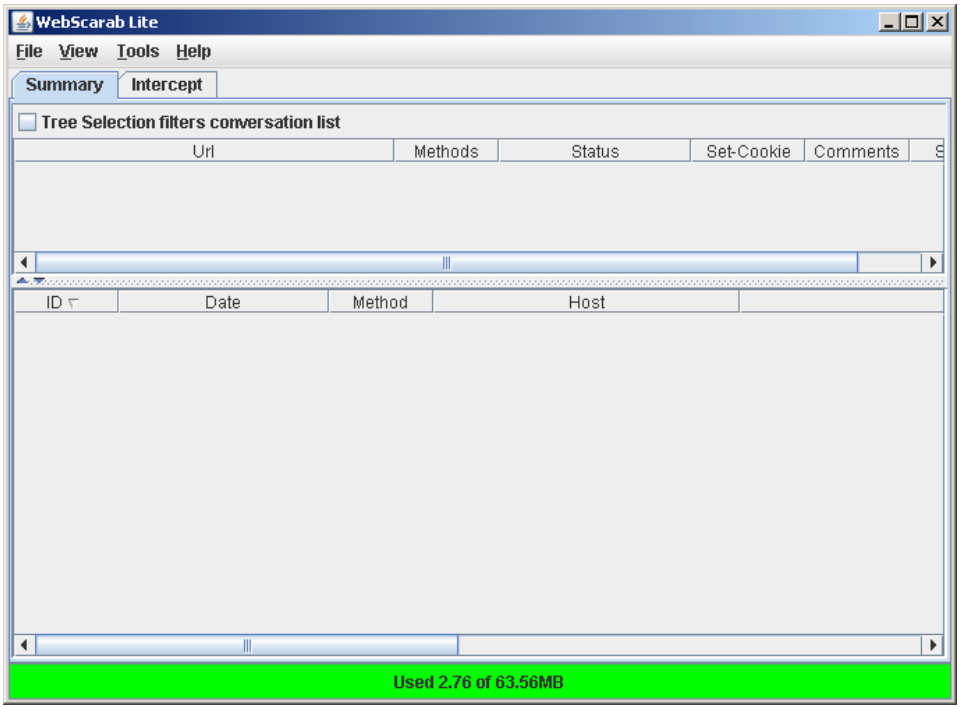
Slika 2. Uključivanje WebScarab posrednog poslužitelja u uobičajenu HTTP komunikaciju

Budući da je za postizanje ovog priključenja klijent potrebno prilagoditi tako da može pristupati posrednom poslužitelju, na slijedećoj je slici dan kratak pregled ovog postupka za Internet Explorer preglednik Windows operacijskih sustava.

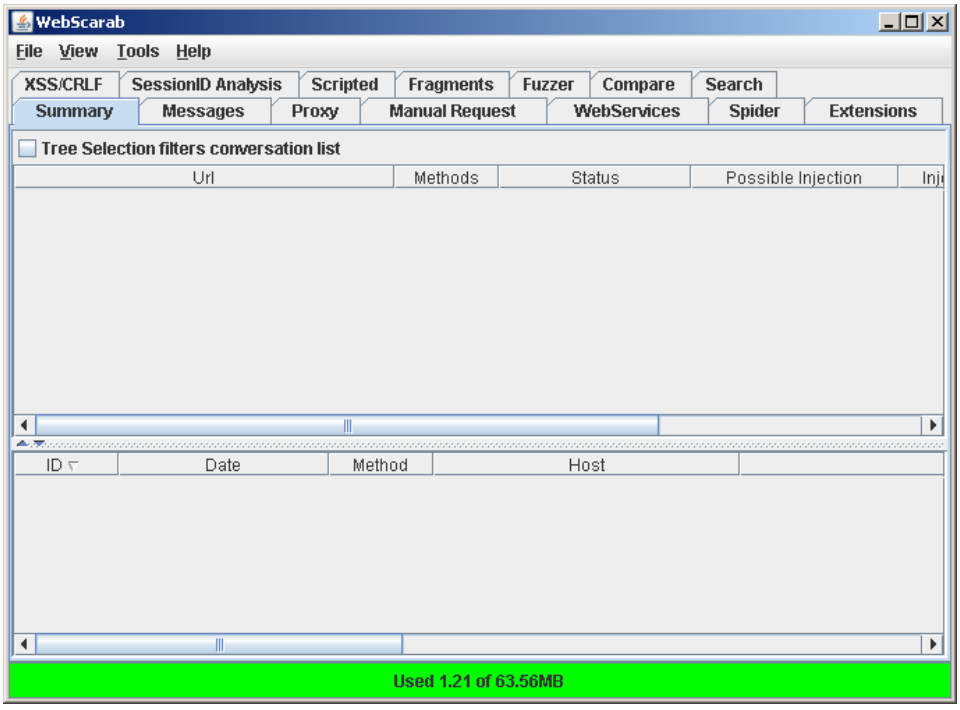


Slika 3. Izmjena postavki klijenta

Prilikom izvođenja ovog postupka potrebno je obratiti pozornost na polja *Address* i *Port*. Ovdje je potrebno unijeti IP adresu i priključak na kojima se nalazi WebScarab posredni poslužitelj.
Također je važno primijetiti kako su opcije *Automatically detect settings* i *Bypass proxy server for local addresses* isključene. Nakon što su prilagođene postavke preglednika i pokrenut WebScarab, moguće je započeti s ispitivanjem prometa između dvije strane (preglednika i poslužitelja).
Postoje dva režima rada sučelja WebScarab alata, prvi (*Lite Interface*) je jednostavniji i namijenjen je manje zahtjevnim korisnicima, dok je drugi potpuno funkcionalan i bitno složeniji. Oba su prikazana na slijedećim slikama.



Slika 4. WebScarab *Lite* sučelje



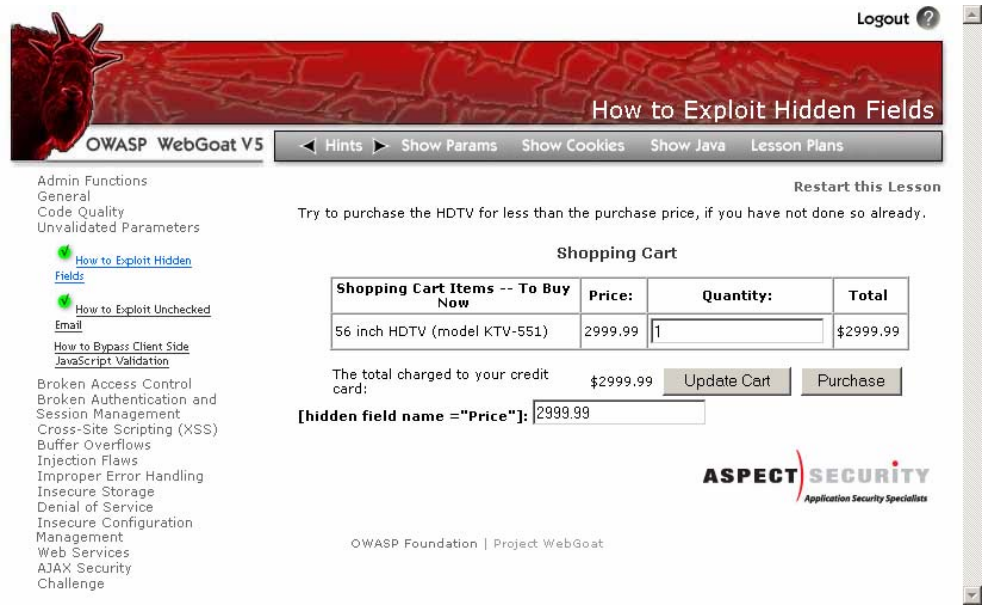
Slika 5. WebScarab *Advanced* sučelje

Kako bi promijenili sučelje u naprednije potrebno je u izborniku *Tools* označiti opciju *Use full-featured interface* te ugasiti i ponovo pokrenuti aplikaciju.

Sučelje WebScarab alata sastoji se od kartica koji predstavljaju različite elemente rada. Na kartici *Summary* mogu se pronaći stabla posjećenih stranica, poslani zahtjevi i primljeni odgovori, a posebno su označeni i odgovori u kojima postoji određena ranjivost.

3.4. Napad na skrivena HTML polja izmjenom HTTP zahtjeva

Skrivena HTML polja su polja koja postoje u HTML stranici, ali nisu izravno prikazana korisniku te ih on ne može jednostavno izmijeniti. Jedna od mogućih uporaba skrivenih polja je prosljeđivanje skrivenih parametara poslužitelju. U slučaju web trgovine to, primjerice, može biti cijena artikla. Slijedi analiza spomenutog scenarija korištenjem mogućnosti WebScarab alata i simuliranog propusta unutar WebGoat web stranice.

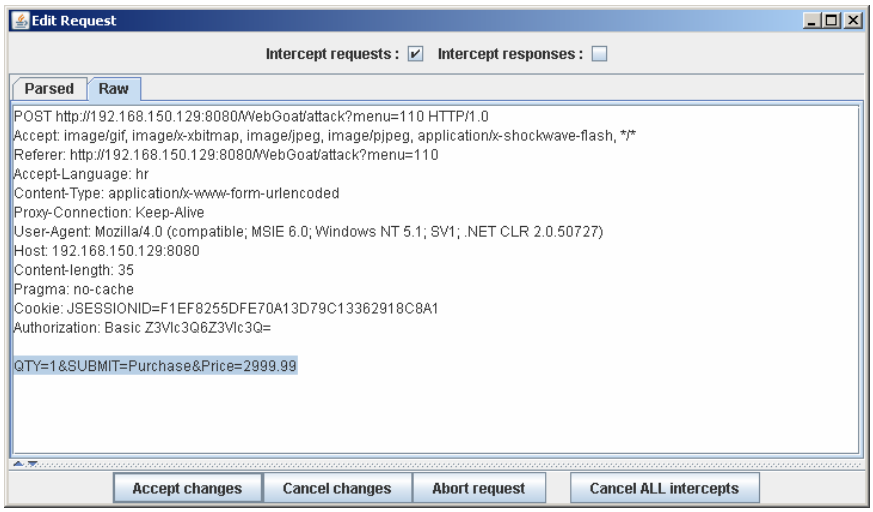


Slika 6. Web stranica sa skrivenim poljem

Prikazana stranica sadrži polje *Quantity* koje označava broj proizvoda koje želimo kupiti te naredbe *Update Cart* i *Purchase*. Prva od njih ažurira stanje košarice, a potonja obavlja akciju kupovine. Na stranici je prikazano i skriveno polje imena *Price* koje je WebScarab primijetio i naznačio.

Kao demonstracija rada WebScarab alata poslužit će primjer u kojem se prikazani artikl kupuje za manji iznos.

Prvi korak u tom postupku je analiza zahtjeva koji stranica šalje poslužitelju prilikom obavljanja naredbe *Purchase*. Kako bi se to učinilo potrebno je u WebScarab kartici *Proxy* označiti polje *Intercept requests*, a u lijevom izborniku plavom bojom označiti POST metodu. Nakon toga potrebno je na web stranici odabrati *Purchase*. WebScarab će signalizirati presretanje poruke novim prozorom imena *Edit Request*. On je prikazan na slijedećoj slici.

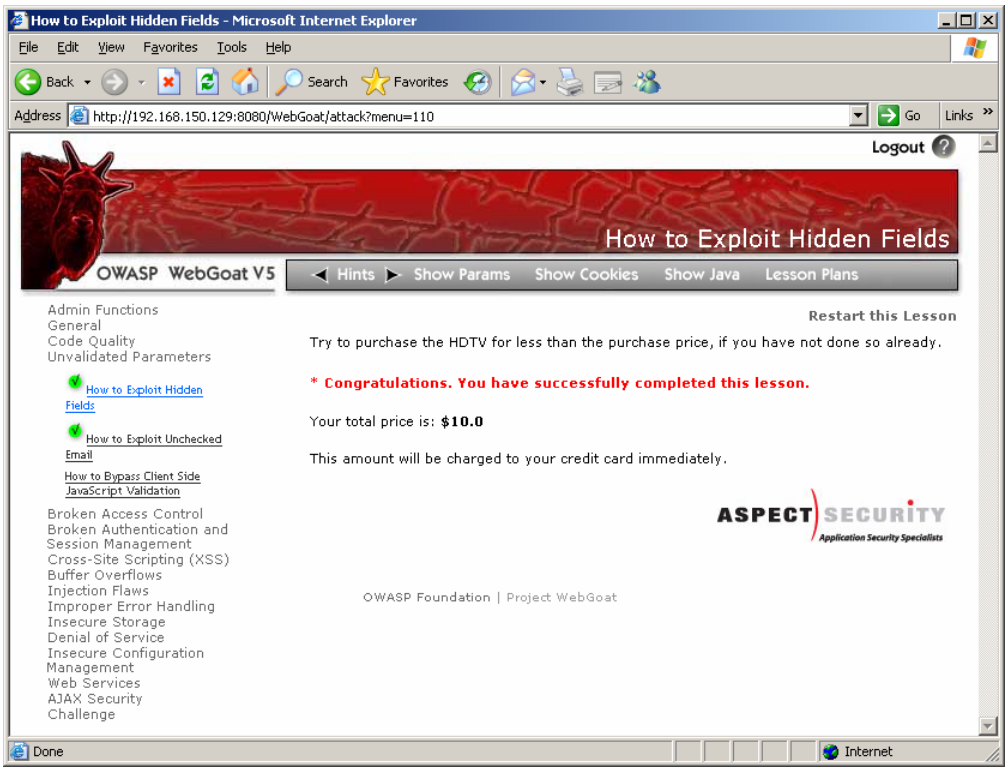


Slika 7. *Edit Request* prozor

Prvi red ispisa predstavlja vrstu zahtjeva koja će se obaviti, u ovom slučaju POST, a nakon toga slijede dodatna polja koja šalje preglednik. Na kraju ispisa se nalaze imena parametara i njihove vrijednosti. Na slici su prikazani slijedeći parametri:

- QTY, odnosno broj artikala (eng. *quantity*),
- SUBMIT, odnosno vrsta akcije se obavlja (u ovom slučaju to je kupnja - eng. *purchase*) te
- PRICE, odnosno cijena artikala.

Jedna od mogućnosti koju pruža WebScarab je izravna izmjena HTTP/HTTPS zahtjeva. Ovdje ju je potrebno iskoristiti te promijeniti vrijednost polja *Price* u, primjerice, 10. Izmjenom vrijednosti i potvrdom na *Accept changes* izmijenjeni zahtjev će se proslijediti poslužitelju koji će ga obraditi i poslati odgovor. Odgovor se prikazuje u pregledniku, što je vidljivo na slijedećoj slici.



Slika 8. Odgovor na izmijenjeni zahtjev

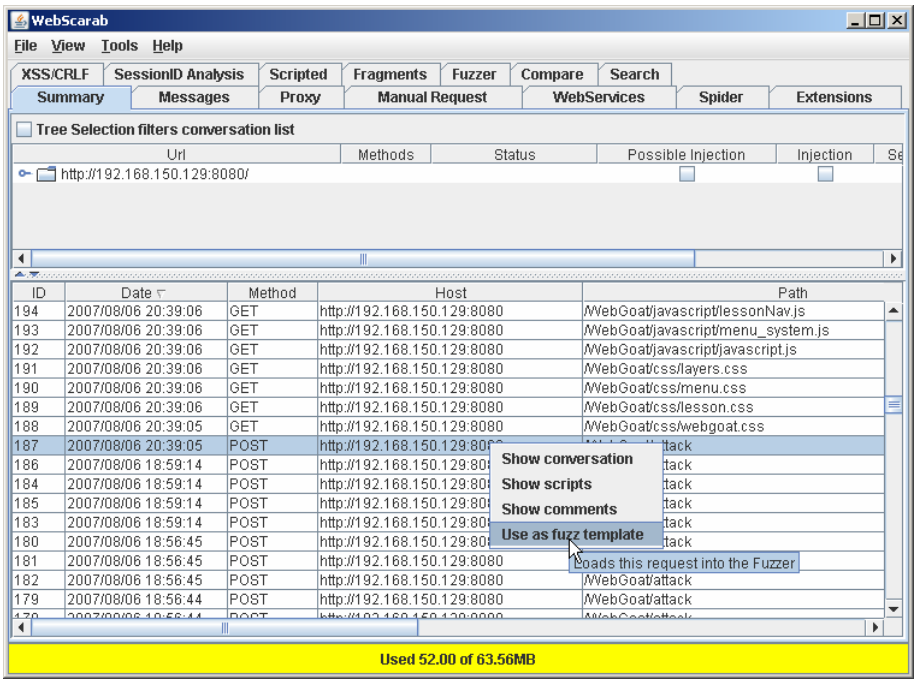
Iz dobivenog odgovora poslužitelja vidljiva je uspješna izmjena vrijednosti polja koje je označavalo cijenu, te ostvarena kupnja HDTV televizora za 10 dolara. Opasnost napada dolazi zbog krive pretpostavke da polja koja nisu vidljiva na stranici nije moguće primijetiti i izmijeniti. Korištenje skrivenih polja za pohranu osjetljivih informacija je loša sigurnosna praksa koju treba izbjegavati, a skrivena polja treba smatrati nepouzdanim.

3.5. Korištenje *fuzzing* modula za detekciju *SQL Injection* propusta

Fuzzing ili *fuzz testing* je način testiranja aplikacija prosljeđivanjem nasumičnih ili posebno oblikovanih argumenata. Tako oblikovanim argumentima pokušavaju se provjeriti rubni slučajevi koje programer možda nije predvidio. Iznimno dugi i kratki argumenti, specijalni znakovi, negativni i veliki pozitivni brojevi samo su neki od njih. Usprkos svojoj jednostavnosti *fuzzing* je izuzetno uspješna tehnika testiranja sigurnosti aplikacije, a vrijeme potrebno za pronalaženje sigurnosnog propusta je mnogo kraće od ručne provjere programskog koda. WebScarab omogućuje testiranje web aplikacija korištenjem predodređenih argumenata iz proizvoljne datoteke. Prilikom testiranja WebScarab će svaki redak odabrane datoteke poslati kao argument odabranim poljima i zabilježiti odgovor poslužitelja. Nakon završetka rada potrebno je ručno provjeriti odgovore poslužitelja.

Postoje dva načina *fuzzing* testiranja parametara. Prvi je ručno kreiranje zahtjeva popunjavanjem odgovarajućih polja unutar *Fuzzer* kartice. Drugi način je korištenje predložaka iz uhvaćenog prometa.

Potonji način je puno pogodniji jer ne zahtjeva dodatnu analizu upućenih zahtjeva i korištenih polja, niti detaljno poznavanje HTTP protokola.
U slijedećem primjeru prikazano je korištenje potonjeg načina za detekciju *SQL Injection* ranjivosti. Kada se pronađe željeni zahtjev u kartici *Summary* potrebno je iznad njega pritisnuti desni gumb miša te odabrati *Use as fuzz template*.

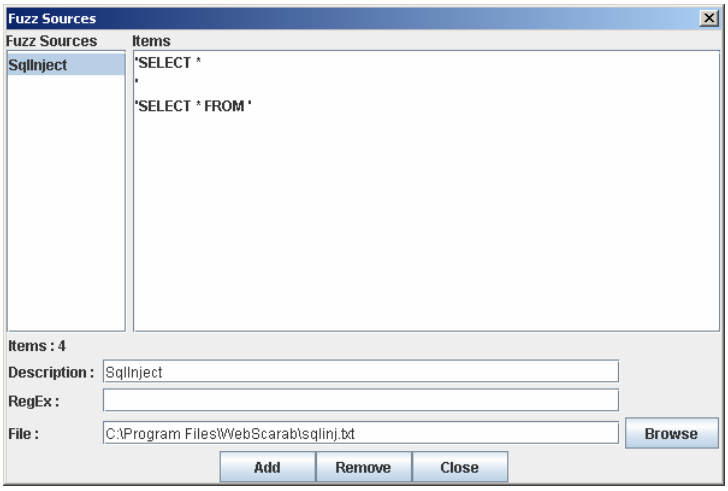


Slika 9. Korištenje WebScarab alata za *fuzzing* testiranje

Nakon što je odabran predložak za testiranje potrebno se prebaciti na *Fuzzer* karticu. Tu su prikazana imena polja, vrste argumenata koji se prosljeđuju parametrima, vrijednosti koje su parametri imali prilikom slanja poslužitelju i sl. Za ispitivanje *SQL Injection* propusta kreirana je datoteka `SqlInject.txt` koja sadrži slijedeće podatke:

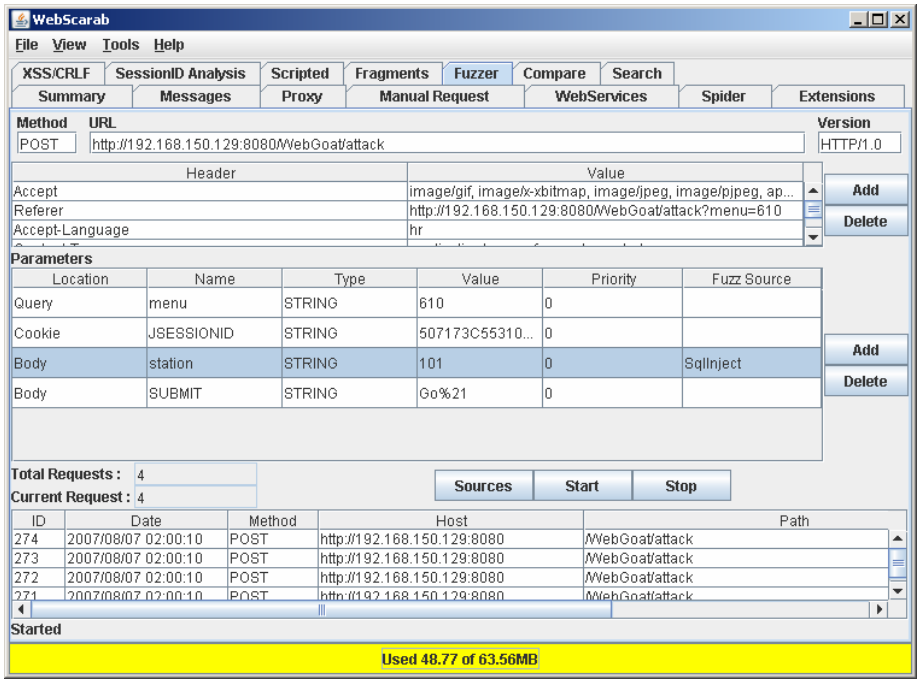
```
'SELECT *
'
'SELECT * FROM'
'SELECT'
```

To su neke od uobičajenih vrijednosti kojima je moguće uočiti *SQL Injection* propust. Svim provjerama zajednički je jednostruki navodnik koji se, prilikom uključivanja parametra u SQL naredbu, interpretira kao njegov završetak te se svi ostali znakovi tretiraju kao SQL izraz. Ovu pojavu napadač može iskoristiti za izmjenu ili razotkrivanje podataka pohranjenih u bazi.
Nakon što je datoteka kreirana potrebno ju je dodati u listu mogućih provjera odabirom *Sources* naredbe te upisom putanje do datoteke i opisa sadržanih provjera. Dijalog koji to omogućuje prikazan je na slijedećoj slici.



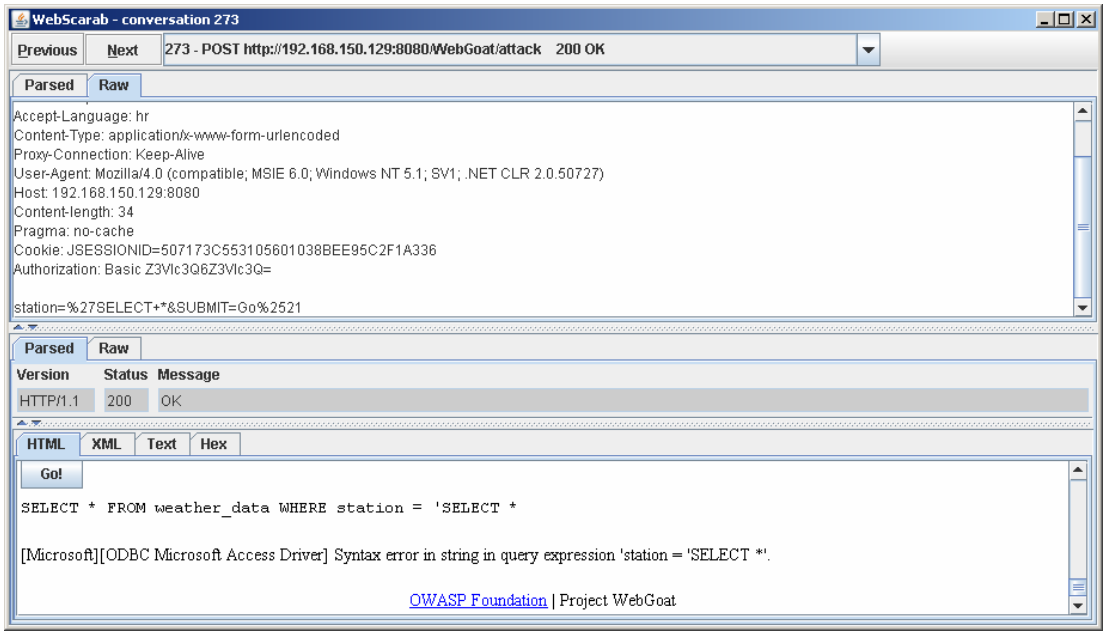
Slika 10. Učitavanje datoteke s popisom *fuzzing* parametara

Dodavanjem mogućih provjera u padajućem su se izborniku „*Fuzz Source*" stupca pojavile odgovarajuće mogućnosti. Za svako polje koje želimo testirati potrebno je odabrati jednu od ponuđenih provjera, te pritiskom na Start započeti testiranje parametara.



Slika 11. Kartica *Fuzzer*

Završetkom provjere na dnu prozora će se pojaviti odgovori poslužitelja. Dvoklikom na željeni odgovor dobiva se rezultat ispitivanja, a u njemu je potrebno pronaći eventualnu grešku obrade SQL upita. Ukoliko se greška pronađe potrebno je poduzeti odgovarajuće mjere zaštite prilikom obrade korisničkih parametara. Na slijedećoj slici prikazan je rezultat ispitivanja koji sadrži SQL pogrešku.



Slika 12. Odgovor koji sadrži SQL pogrešku

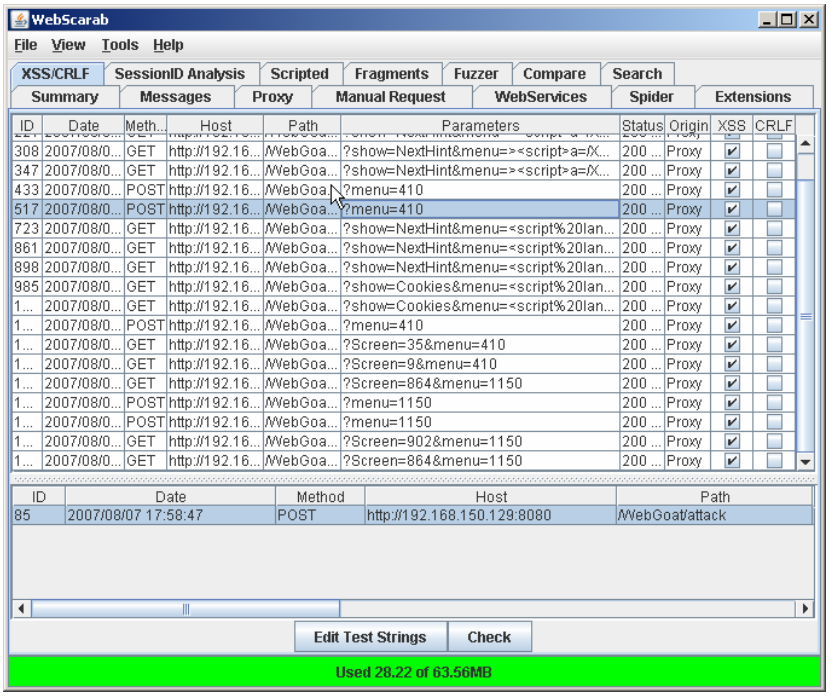
3.6. Korištenje XSS/CRLF modula za detekciju XSS i CRLF ranjivosti

XSS/CRLF modul analizira sumnjive HTTP zahtjeve tražeći tzv. XSS i CRLF ranjivosti. Ranjivi zahtjevi se prikupljaju isključivo pasivnom analizom svih HTTP sjednica koje prolaze kroz WebScarab. Modul će ispitati svaki zahtjev i odgovor te provjeriti da li je:

- bilo koja vrijednost GET/POST parametra reflektirana u tijelu HTTP odgovora, što signalizira potencijalnu XSS ranjivost, te da li je
- bilo koja vrijednost GET/POST parametra reflektirana u zaglavlju HTTP odgovora, što signalizira potencijalnu CRLF ranjivost.

Svi potencijalno ranjivi zahtjevi biti će prikazani u XSS/CRLF kartici uz ranjivost naznačenu u odgovarajućem stupcu.

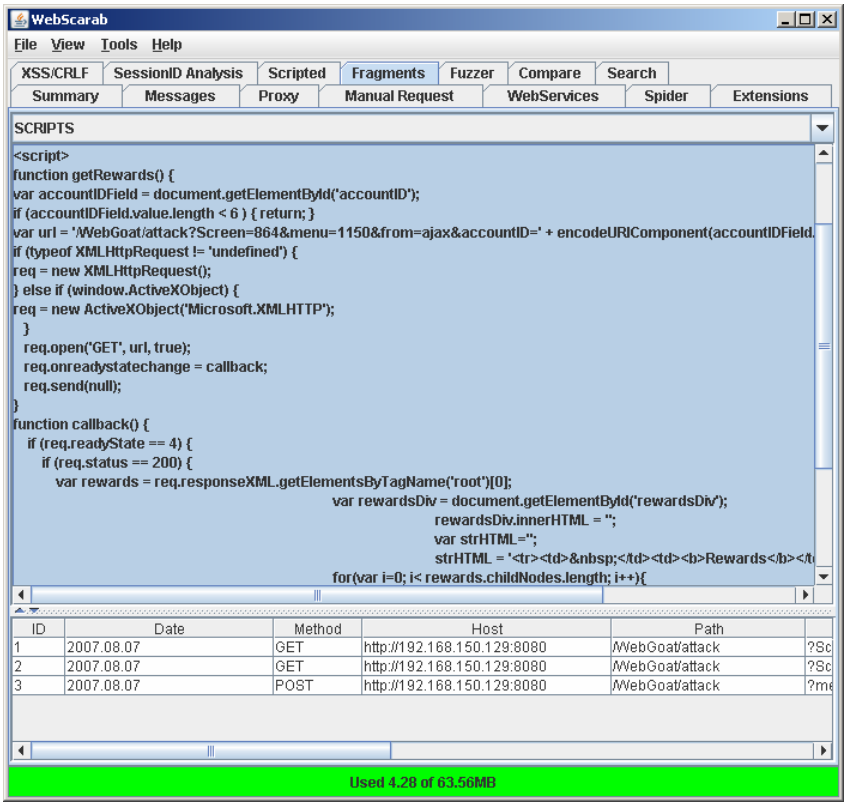
Važno je primijetiti da su ispisane samo potencijalne ranjivosti te da je svaku od njih potrebno potvrditi. Kako bi se to učinilo potrebno je odabrati željeni zahtjev te *Check* dugme. Ukoliko je zahtjev ranjiv pojavit će se u donjem prozoru zajedno s parametrima kojima je testiran propust.



Slika 13. XSS/CRLF kartica

3.7. Pregledavanje sakupljenih skripti

Još jedna od mogućnosti WebScarab alata je i automatsko izdvajanje i prikupljanje skripti i komentara iz HTML koda. WebScarab će pasivnom analizom HTTP odgovora detektirati pojavljivanje komentara i skripti koje je naknadno moguće pregledavati unutar *Fragmets* kartice.

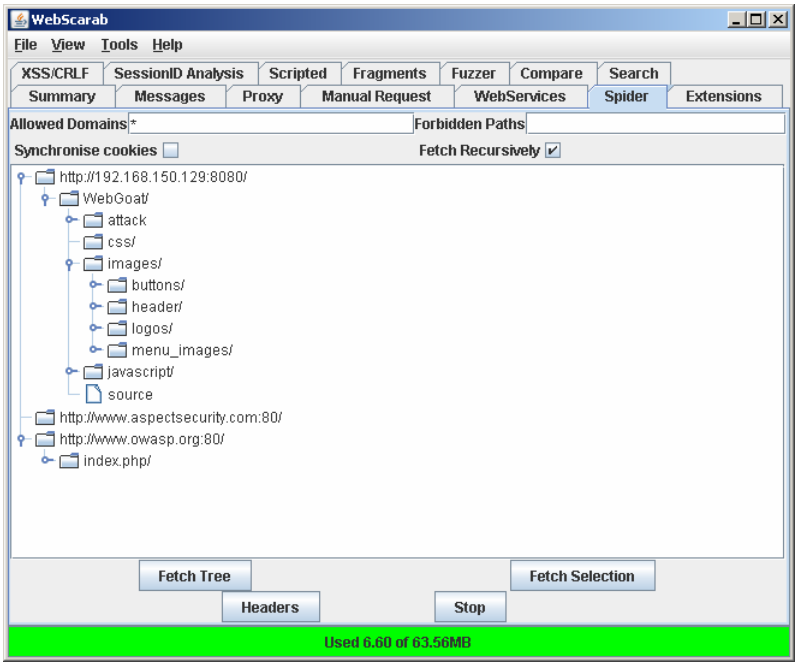


Slika 14. *Fragments* kartica

Uz svaki komentar i skriptu zabilježeni su i odgovarajući zahtjevi koji su ih generirali, što korisniku omogućuje lakše razumijevanje sadržaja skripte i parametara na koje se odnosi.

3.8. *Spider* modul

Spider je modul koji omogućuje prikupljanje podataka o strukturi datoteka i direktorija na poslužitelju. Prilikom posjete web stranice sadržaj se dohvaća iz raznih datoteka koje su uključene u HTML kod. Spomenute adrese *Spider* može iskoristiti za kreiranje stabla direktorija i datoteka te ih prikazati. Prilikom korištenja Proxy modula *Spider* će automatski dodati sve datoteke kojima se pristupalo i izgraditi početno stablo.

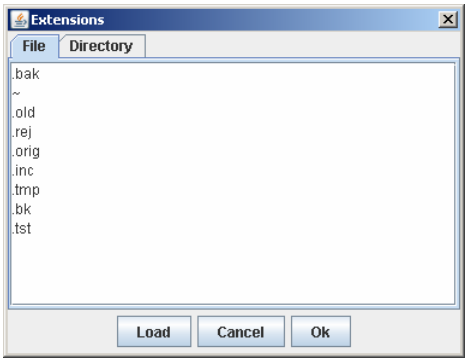


Slika 15. *Spider* kartica

Postojeće stablo je moguće proširiti dodatnim podacima koristeći rekurzivni postupak koji će uporabom dosad sakupljenih podataka pokušati dohvatiti sve referencirane stranice i sadržaje. Dodatne restrikcije na mjesta s kojih se dohvaćaju podaci moguće je odrediti upisom odgovarajućih adresa u *Allowed Domains* polje. Zvjezdica omogućava dohvaćanje svih datoteka, što može rezultirati velikim brojem rezultata ukoliko je web stranica povezana s velikim brojem vanjskih resursa. Pokretanjem naredbe *Fetch Tree* započinje postupak dohvaćanja podataka i izgradnje stabla.

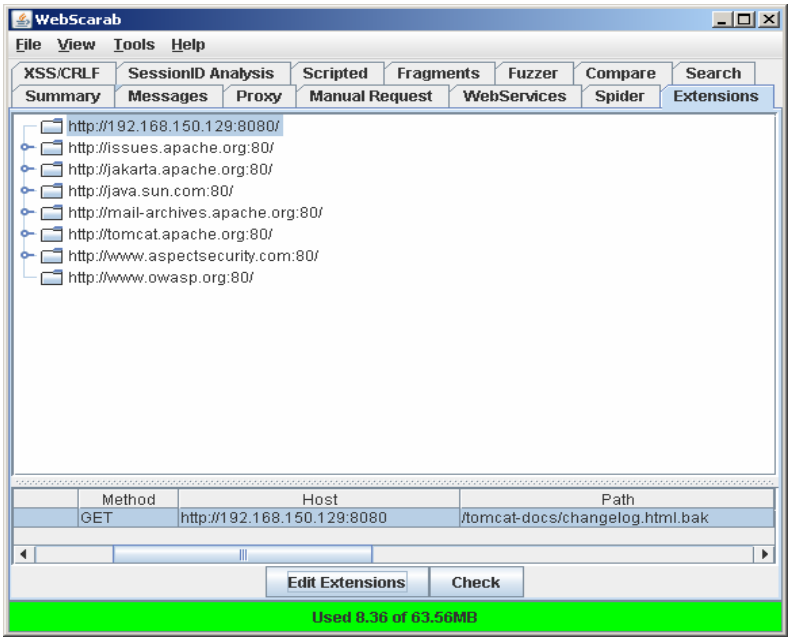
3.9. Provjera zaostalih datoteka

Prilikom izrade web stranice i izmjene dokumenata neki uređivači teksta spremaju stare inačice datoteke s nastavcima poput `.bak`, `.old`, `~`, `.tmp` i sl. WebScarab sadrži modul koji omogućava pretragu podataka sakupljenih Proxy ili Spider modulom za predodređenom listom nastavaka datoteka. Osim spomenutih, moguće je tražiti i datoteke nastale izradom sigurnosne kopije direktorija. One uobičajeno imaju nastavke poput `.zip`, `.tar`, `.bz2`, `.bk`, `.exe` i td. Sve željene nastavke moguće je dodati u listu odabirom *Edit Extensions* naredbe.



Slika 16. Zadavanje nastavaka datoteka

Oznakom željene adrese web poslužitelja i odabirom *Check* naredbe moguće je započeti provjeru njegovih zaostalih datoteka i direktorija.

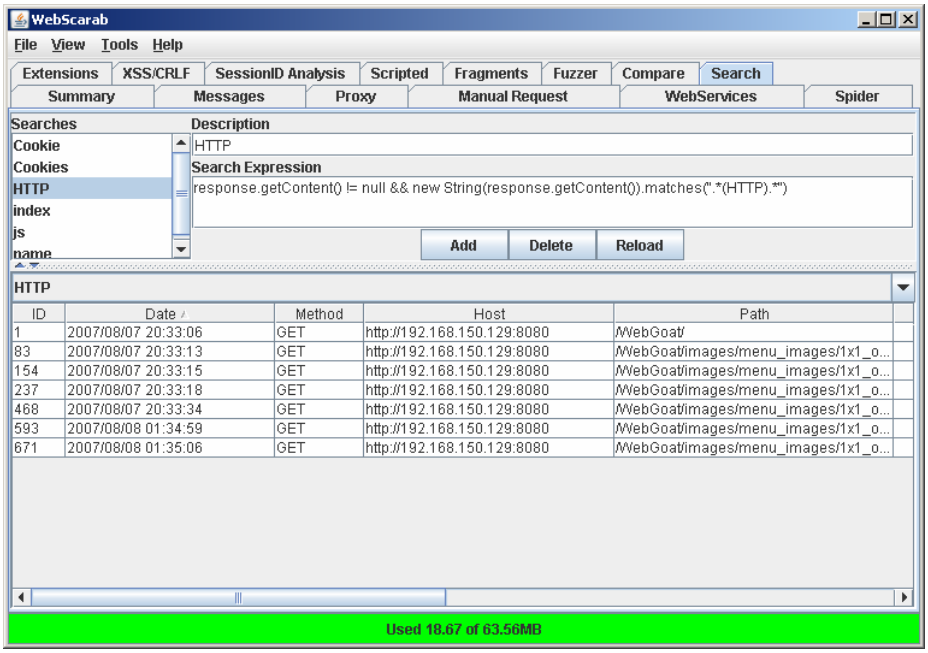


Slika 17. *Extensions* kartica

Pronađene datoteke biti će ispisane zajedno s putanjom, zahtjevom kojim je provjereno njihovo postojanje, adresom poslužitelja na kojoj su pronađene i datumom provjere.

3.10. Search modul

Ovaj modul omogućuje naprednu pretragu zaglavlja i tijela unutar zahtjeva i odgovora zabilježenih Proxy modulom. Pretraga se obavlja pisanjem odgovarajućeg *BeanShell* programskog koda koji sadrži regularni izraz za pronalazak željenog teksta.

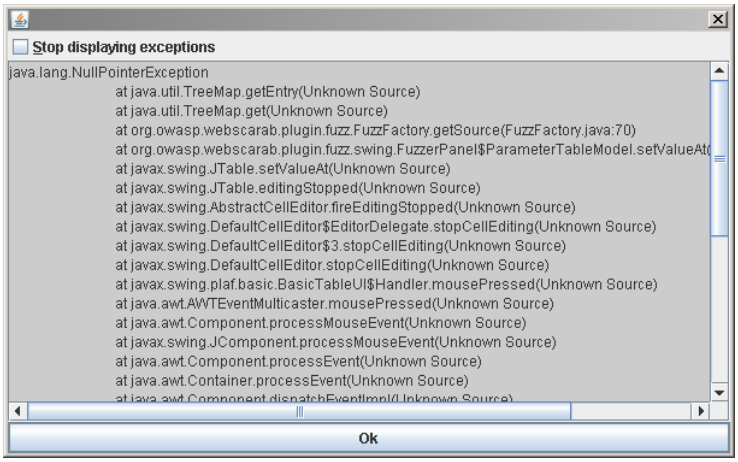


Slika 18. *Search* kartica

Korištenjem BeanShell jezika i javnih WebScarab funkcija moguće je konstruirati pretragu koja će pronaći čitave razgovore preglednika i poslužitelja koji zadovoljavaju skup uvjeta. Svakoj od pretraga dodijeljeno je ime pomoću kojega je moguće pristupati rezultatima pretrage. Pretraga započinje pokretanjem *Add* naredbe i sve naknadno zabilježene sjednice automatski će biti provjerene te, ukoliko zadovoljavaju uvjete, dodane u listu pronađenih sjednica.

3.11. Uočeni nedostaci

Prilikom korištenja WebScarab alata nekoliko je puta došlo do rušenja i blokiranja programa nakon čega je bilo potrebno ugasiti ga i ponovno pokrenuti. Također je primjetno i usporavanje programa tijekom pristupanja poslužitelju odnosno korištenja modula za pretragu, što je uočljivo čak i na manjem broju zabilježenih sjednica. Kako bi se spriječila učestalo pojavljivanje greške korištenja NULL pokazivača ponuđena je i opcija zanemarivanja daljnjeg prijavljivanja greške.



Slika 19. Pogreška pojave NULL pokazivača koja rezultira rušenjem alata

4. Zaključak

Budući da se s vremenom povećava broj zlouporaba različitih sigurnosnih propusta, povećava se i svijest o potrebi odgovarajuće zaštite.
OWASP je jedan od projekata koji pokušava sustavno utjecati na korisnike i povećati stupanj njihovog razumijevanja računalne sigurnosti. Osim toga nudi i alate za samostalno praćenje i povećanje sigurnosti vlastitih aplikacija. Jedan od takvih alata je i WebScarab. On je namijenjen web programerima i ostalim zainteresiranima za sigurnost web aplikacija. Kroz jednostavno sučelje, s mnoštvom različitih modula, omogućava testiranje velikog spektra web aplikacija i sigurnosnih propusta vezanih uz njih. Poštivanjem tradicije otvorenog koda WebScarab korisnicima pruža veliku slobodu omogućujući tako dodavanje novih modula i funkcionalnosti. Mnoštvo propusta u stabilnosti ne čini ga pogodnim za korištenje na velikim projektima, ali je u kombinaciji s projektima poput onog nazvanog *WebGoat* vrlo dobro sredstvo za razumijevanja kako tehničke pozadine raznih sigurnosnih propusta, tako i različitih mogućnosti njihove zlouporabe.

5. Reference

[1] OWASP Project, http://www.owasp.org/index.php/Category:OWASP_Project, srpanj 2007.
[2] WebScarab, http://www.owasp.org/index.php/Category:OWASP_WebScarab_Project, srpanj 2007.
[3] WebGoat, http://www.owasp.org/index.php/Category:OWASP_WebGoat_Project, srpanj 2007.
[4] WebScarab, http://dawes.za.net/rogan/webscarab/docs/, srpanj 2007.
[5] HTTP response splitting, http://en.wikipedia.org/wiki/Http_response_splitting, srpanj 2007.