HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Brute force napadi

CCERT-PUBDOC-2007-08-201

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 4

2. OPĆENITO O *BRUTE FORCE* NAPADIMA 5

2.1. IMPLEMENTACIJA *BRUTE FORCE* NAPADA 5
2.2. OPTIMIZACIJE *BRUTE FORCE* NAPADA 6
2.3. NAPAD RJEČNIKOM 6
2.4. PROBOJI KRIPTOGRAFSKIH ALGORITAMA 7

3. DETEKCIJA I ZAŠTITA OD *BRUTE FORCE* NAPADA 8

3.1. DETEKCIJA I ZAŠTITA OD *BRUTE FORCE* NAPADA NA WINDOWS WEB POSLUŽITELJU 9

4. TESTIRANJE RANJIVOSTI NA *BRUTE FORCE* NAPADE 11

5. ZAKLJUČAK 13

6. REFERENCE 13

1. Uvod

Brute force napad ili napad uzastopnim pokušavanjem je jednostavna, ali uspješna tehnika rješavanja problema koja se sastoji od sustavnog pronalaženja svih mogućih kandidata za rješenje i isprobavanja svakog od njih. *Brute force* napad je jednostavan za implementaciju i često se koristi za probijanje zaporki ili raznih enkripcija, a osim toga, on uvijek i pronalazi rješenje, ako ono postoji. Međutim, vrijeme i resursi potrebni za rješavanje problema ovom metodom rastu proporcionalno broju mogućih kandidata za rješenje. Zbog toga se algoritam napada često optimizira heurističkim dodacima koji bitno umanjuju broj mogućih kandidata. Sami napadi i metode za njihovu optimizaciju opisani su u nastavku dokumenta, a uz njih je dan i pregled proboja nekih poznatih algoritama koji su postignuti korištenjem ovog pristupa. Osim toga dokument opisuje i načine detekcije i zaštite protiv *brute force* napada, te daje i kratke upute o mogućem korištenju *brute force* alata za testiranje ranjivosti vlastite mreže.

2. Općenito o *brute force* napadima

Brute force napadi se također nazivaju i iscrpno pretraživanje jer se temelje na apsolutnom i ravnopravnom pretraživanju i isprobavanju svih mogućih kombinacija. Takvi napadi očito ne mogu biti efikasni, tj. moraju zahtijevati ili velike resurse za isprobavanje velikog broja različitih kombinacija ili veliku količinu vremena dovoljnu da se s manjim resursima provedu isprobavanja svih kombinacija. U stvarnoj primjeni to znači da je cijena resursa potrebnih za rješavanje nekog problema ovom metodom često veća nego što vrijedi samo rješenje. Ako se, primjerice, radi o probijanju zaporke to znači da su resursi potrebni za probijanje vrjedniji od onog do čega se potencijalno može doći nakon probijanja zaporke.

Brute force napad ne treba zamijeniti s drugim sličnim vrstama napada koje raznim metodama reduciraju broj mogućih rješenja za isprobavanje. *Brute force* metodom isprobavaju se sva moguća rješenja bez obzira na nelogičnost nekih od njih. Na primjer ako se uzme klasični problem 8 kraljica u kojem je potrebno postaviti 8 kraljica na šahovsku ploču, ali na takav način da nijedna od njih ne napada drugu, *brute force* tehnika će za rješenje problema isprobati svih 64! / 56! = 178,462,987,637,760 mogućih rješenja. Nekom drugom metodom kao što je npr. *backtracking* broj ispitanih kandidata prije pronalaska rješenja bio bi znatno manji. Naime, *backtracking* je derivacija *brute force* tehnike koja se može primijeniti u situacijama kada su ispitivani scenariji donekle povezani. Primjerice, u slučaju problema osam kraljica dva razmještaja na ploči mogu biti povezana kraljicama koji stoje na jednakim poljima. Ukoliko se u jednom razmještaju dvije kraljice napadaju, u svim njemu sličnim slučajevima nije potrebno provoditi ispitivanje, jer će se i u njima dogoditi jednaka situacija. Tako je moguće izbjeći stvarno ispitivanje velikog broja kombinacija.

2.1. Implementacija *brute force* napada

Brute force napad je jednostavan za implementaciju i sastoji se od četiri osnovne procedure:

1. `prvi(P)` – generiraj prvi kandidat k za problem P.
2. `slijedeći(P, k)` – generiraj slijedeći kandidat k za problem P nakon trenutnog kandidata k.
3. `važeći(P, k)` – provjeri je li kandidat k važeće rješenje problema P.
4. `izlaz(P, k)` – upotrijebi rješenje k za problem P kao prikladno rješenje za aplikaciju.

Procedura `slijedeći(P, k)` trebala bi također detektirati situacije kad se iscrpe svi kandidati i kao rezultat poslati neku vrijednost (npr. *null*) koja bi omogućila prekid daljnjeg izvođenja algoritma. Jednako bi tako procedura `prvi(P)` trebala vratiti *null* vrijednost ako ne postoji nijedan kandidat za rješenje problema. Algoritam se tako može prikazati slijedećim pseudokodom:

```
prvi (P)
dok je k različito od null radi
   ako je važeći (P, k) tada izlaz (P, k)
   inače slijedeći (P, k)
```

Na primjer kod rješavanja problema pronalaska djelitelja broja n, P poprima vrijednost broja n. Procedura `prvi(P)` vraća prvi važeći kandidat k, a to je broj 1 ako broj 1 nije veći od n ili *null* ako je odabrani broj n = 1. Procedura `slijedeći(P, k)` vraća prvi slijedeći kandidat koji ima vrijednost k + 1 ako je k + 1 < n ili *null* ako je k + 1 > n. Procedura `važeći(P, k)` vraća vrijednost ISTINA samo ako je k djelitelj broja n. Dodatno algoritam se može pojednostavniti ako se umjesto *null* koristi vrijednost n+1.

Brute force algoritam će pozivati proceduru izlaz za svaki generirani kandidat, ali ga se može lako modificirati da prestane s izvođenjem nakon pronalaska prvog rješenja ili određenog broja rješenja. Također moguće ga je modificirati tako da prestane s izvođenjem nakon određenog broja kandidata ili nakon što je izvođenjem potrošena određena količina procesorskog vremena.

Glavni je nedostatak *brute force* metode postojanje izrazito velikog broja mogućih kandidata za veliku većinu stvarnih problema. Na primjer, ako ovim algoritmom tražimo djelitelje broja n, algoritam će isprobati n kandidata. Ako je n šesnaesteroznamenkasti broj isprobavanje zahtijeva izvršavanje

najmanje 10^{15} instrukcija što bi na uobičajenom računalu potrajalo nekoliko dana. Ako je n neki 64-bitni broj koji u prosjeku ima 19 decimalnih znamenki isprobavanje ovim algoritmom ispitivanje bi potrajalo oko 10 godina.
Ovaj nagli porast broja kandidata s porastom veličine podataka se pojavljuje prilikom rješavanja svih vrsta problema. Na primjer, ako se traži određeni raspored 10 slova tada postoji 10! = 3,628,800 mogućih kandidata koje će tipično računalo isprobati unutar jedne sekunde. Međutim ako dodamo još jedno slovo, što predstavlja samo 10%-no povećanje veličine podataka, broj kandidata povećava se 11 puta ili 1000 %. Za 20 slova broj kandidata iznosi 2,4 x 10^{18} i isprobavanje svih njih potrajalo bi oko 10,000 godina. Ovaj fenomen drastičnog porasta broja mogućih kombinacija naziva se još i kombinatorna eksplozija i glavni je razlog zbog kojeg je nužno optimizirati *brute force* metodu da bi ona bila primjenjiva.

2.2. Optimizacije *brute force* napada

Glavni problem *brute force* metode je prevelik broj mogućih kandidata pa je prva logična optimizacija ona kojom se pokušava smanjiti broj kandidata korištenjem heuristike ili mehanizama učenja.
Na primjer ako pogledamo prije spomenuti problem s 8 kraljica koji ima 178,462,987,637,760 mogućih rješenja. Ako uzmemo u obzir činjenicu da su sve kraljice identične i da dvije kraljice ne mogu biti postavljene na isto polje proizlazi da su mogući kandidati sve kombinacije 8 polja na ploči od 64 polja ili ukupno 64! / 56! / 8! = 4,426,165,368 mogućih kandidata.
Daljnjom upotrebom logike može se zaključiti da nijedna kombinacija u kojoj se dvije kraljice nalaze u istom redu ili stupcu na ploči ne može biti rješenje. To znači da je broj kandidata reduciran na sve kombinacije u kojima se svaka kraljica nalazi u svom retku i u bilo kojem stupcu. Takve kombinacije mogu se opisati poljem od 8 brojeva čije vrijednosti mogu biti u rasponu od 1 do 8 gdje vrijednost broja predstavlja stupac, a redni broj člana polja redak u kojem se kraljica nalazi. Iz toga je vidljivo da je broj mogućih kandidata jednak broju svih permutacija vrijednosti članova polja ili brojkom 8! = 40,320 mogućih kandidata, tj. 1/100,000 izvornog broja mogućih kandidata.
Iz ovog je vidljivo da se čak i jednostavnom analizom može drastično smanjiti broj mogućih kandidata i pretvoriti kompleksan problem u znatno jednostavniji. Također vidljivo je da procedure za odabir kandidata (`prvi` i `slijedeći`) čak i za ovakav reducirani skup kandidata nisu složene, nego mogu biti i jednostavnije od izvornih.
U nekim situacijama potrebno je naći samo jedno od nekoliko mogućih rješenja. U tom slučaju postaje bitan redoslijed pronalaska rješenja iz čega proizlazi druga moguća optimizacija kojom se prvo pokušava pronaći najvjerojatnija rješenja, a zatim, ako ona ne zadovoljavaju, ostala rješenja. Tako je na primjer u traženju djelitelja broja n bolje isprobavati kandidate počevši od broja 2 do broja n-1 nego obrnuto jer je vjerojatnost da je n djeljiv s k jednaka 1/k.
Također, vjerojatnost da je neki kandidat rješenje često ovisi o ishodu isprobavanja prethodnih kandidata. Primjer toga je situacija u kojoj se želi pronaći bit vrijednosti 1 u nizu od 1000 bitova P. Mogući kandidati su sve pozicije u nizu od 1 do 1000, a kandidat k je rješenje ako je P(k) = 1. Ako je, na primjer, vjerojatnost kombinacija u kojima je prvi bit 0 ili 1 jednaka i ako je vjerojatnost jednakosti svakog bita s njegovim sljedbenikom 90 %, onda će, u slučaju kada se kandidati ispituju od prvog prema zadnjem, prosječan broj isprobanih kandidata prije pronalaska rješenja biti oko 6. Ako se pak kandidati ispituju redoslijedom 1, 11, 21, 31,...991,2, 12, 22, 32,...prosječan broj isprobanih kandidata prije pronalaska rješenja bit će samo nešto više od 2.
Općenito, strategija bi trebala biti takva da se kao slijedeći uvijek uzima kandidat za koji je vjerojatnost rješenja najveća, uzevši pritom u obzir neuspjeh prethodnih pokušaja. Drugim riječima, ako su rješenja donekle grupirana, svaki novi kandidat treba biti što dalje od prethodnih ili obrnuto – ako su rješenja distribuirana jednoliko, onda je optimalna strategija odabira kandidata što bližeg prethodnom.

2.3. Napad rječnikom

Napad rječnikom (eng. *dictionary attack*) je tehnika za probijanje šifri ili autentikacije kod koje se pokušava pronaći tražena zaporka ili tajni ključ uzastopnim isprobavanjem velikog broja riječi ili kombinacija riječi. Napad rječnikom je varijacija *brute force* napada kod koje je izbor mogućih kandidata sužen sa skupa svih mogućih kombinacija slova na one kombinacije koje imaju neko

značenje na određenom jeziku. Izbor je logičan jer je veća vjerojatnost da je zaporka ili ključ neki skup slova koji ima značenje nego neki niz znakova koji korisniku ništa ne znači i koji mu je zbog toga teško pamtljiv.
Riječi za napad rječnikom se uzimaju iz rječnika koji su dostupni na Internetu i ne predstavljaju potpun rječnik određenog jezika već uključuju veći ili manji skup najvjerojatnijih riječi, tj. riječi koje se najčešće koriste u zaporkama. Budući da ljudi pokazuju sklonost odabiru zaporki od 7 ili manje znakova koje su, k tome, još i riječi s nekim značenjem ili neki predvidivi anagrami / skraćenice ili pak riječi nadopunjene jednoznamenkastim brojem napadi rječnikom su prilično uspješni.
Napadi rječnikom se uglavnom upotrebljavaju u dvije svrhe:

- U kriptografskoj analizi za otkrivanje ključa za dekripciju određenog teksta, te
- Za probijanje autentikacije, tj. otkrivanje zaporke nekog sustava sa svrhom ostvarenja neovlaštenog pristupa.

Postoji i treća neizravna primjena napada rječnikom koju koriste autori neželjenih poruka. Naime oni koriste rječnike za generiranje adresa elektroničke pošte koje koriste za slanje neželjenih poruka (eng. *e-mail address harvesting*). Primjerice, autori poruka generirat će poruke za adrese: adam@example.com, barbara@example.com, carl@example.com, itd. Ako na koju od tih adresa poruka bude dostavljena, tj. ako se ne dobije poruka o nemogućnosti dostave autor neželjenih poruka je bilježi kao važeću i koristi za slanje daljnjih neželjenih poruka.

2.4. Proboji kriptografskih algoritama

Brute force napadi se često koriste za probijanje kriptografskih algoritama, pa se čak i pojam probijanja kriptografskog algoritma definira kao pronalazak metode za proboj efikasnije od *brute force* metode. Zbog toga se većina kriptografskih algoritama i dizajnira tako da se onemogući proboj *brute force* metodom, tj. dizajn mora biti takav da bi za njegov proboj *brute force* metodom trebalo previše vremena s trenutno raspoloživim računalnim resursima.
Brute force napad na kriptografski algoritam je potraga za ključem pomoću kojeg će se iz nekog zaštićenog teksta dobiti njegov izvoran oblik. Statistički, ako se ključevi biraju potpuno slučajno, izvoran tekst će biti probijen tek nakon što je isprobano pola mogućih ključeva, naravno uz pretpostavku da je algoritam kriptiranja poznat. To je ujedno i temeljna postavka kriptografije – sigurnost mora ležati u ključu jer napadač uvijek poznaje algoritam.
Od 2002. godine simetrične algoritme s ključevima dugim do 64 bita moguće je probiti *brute force* napadima. DES algoritam koji koristi 56-bitni ključ probila je EFF grupa krajem 90-ih godina. Budući da je EFF kao neprofitna udruga uspjela ostvariti proboj, logično je pretpostaviti da neka komercijalna institucija kojoj su dostupna znatno veća sredstva to vrlo lako može ponoviti. Zbog toga stručnjaci preporučuju upotrebu minimalno 128-bitnog ključa i to uz upotrebu nekog sigurnog algoritma. Naime mnogi algoritmi mogu biti reducirani na efektivnu dužinu ključa koja znatno smanjuje vrijeme potrebno za provedbu *brute force* napada.
Kod asimetričnih algoritama situacija je nešto složenija i uvelike ovisi o algoritmu o kojem se radi. Tako je trenutno moguće probiti RSA zaštitu ključem duljine 512 bita, dok je većinu algoritama temeljenih na eliptičnim krivuljama moguće probiti ako koriste ključ dužine 109 bita (probijen javno 2003. godine). Trenutne preporuke za dužine ključeva su 128 bita za eliptične algoritme i 1024 bita za RSA algoritme. Činjenica je da će algoritme koji sigurnost temelje na složenim matematičkim problemima uvijek biti lakše probiti jer se gotovo uvijek mogu pronaći neke prečice u izračunu. Zbog toga se kod ovih algoritama preporuča upotreba dužih ključeva.
U nastavku je dan popis nekih kriptografskih algoritama i vrijeme njihova proboja:

- RC4 40-bitni ključ – probijen 1995. godine.
- RC5 48-bitni ključ – probijen 1997. godine.
- RC5 56-bitni ključ – probijen 1997. godine.
- RC5 64-bitni ključ – probijen 2002. godine (trajanje1757 dana).
- Eliptični 109-bitni ključ – probijen 2000. godine.
- DES 64-bitni – probijen 1997. godine.
- RSA 512-bitni – probijen 1999. godine.

Općenito se smatra da je 128 bitni ključ dovoljno siguran od *brute force* napada kod simetričnih algoritama. Naime u slučaju 128-bitnog ključa svaka od 2^{128} kombinacija mora biti isprobana. Ukoliko

bi to radilo računalo koje može isprobati 10^{18} kombinacija u sekundi bilo bi potrebno oko 10^{13} godina da se isprobaju sve kombinacije. Usporedbe radi svemir postoji oko 10^{10} godina. Istina rješenje bi vjerojatno bilo pronađeno nakon što bi se isprobalo pola od mogućih kombinacija, ali, obzirom na red veličina u kojem je izraženo potrebno vrijeme, to ne igra veliku ulogu.
Treba napomenuti da se probijanje algoritama temelji na pretpostavci mogućnosti raspoznavanja točnog rješenja. Ako izvoran tekst nije poznat onda se dobiveno rješenje uspoređuje s riječima iz rječnika kako bi se utvrdilo je li smisleno ili nije. To koriste i neke metode zaštite te zagađuju izvoran tekst prije nego ga podvrgnu enkripciji. U takvoj situaciji čak i ako napadač uspije dešifrirati tekst, rješenje će biti nešto što nije smisleno pa ga neće prepoznati.

3. Detekcija i zaštita od *brute force* napada

Brute force napadi se obično provode na jedan od slijedeća tri načina:

1. ručnim pokušajima autentikacije uzastopnim unošenjem korisničkih imena i zaporki,
2. napadima rječnikom pomoću automatiziranih skripti i programa koje unose tisuće raznih kombinacija korisničkih imena i zaporki iz rječničkih datoteka, te
3. generiranim korisničkim računima – zlonamjerno oblikovanim programom generiraju se slučajna korisnička imena i zaporke pomoću kojih se pokušava pristupiti sustavu.

Bilo koji od ovih napada se lako može uočiti pregledom dnevničkih (eng. *log*) zapisa računala. Ako u dnevniku postoje zapisi o velikom broju neuspješnih pokušaja autentikacije onda se vrlo vjerojatno radi o *brute force* napadu. Pregledom dnevnika potrebno je tražiti zapise slične ovom:

```
Apr 11 19:02:10 fox proftpd[6950]: poslužitelj
(usersip[usersip]) - USER korisničkoime (Login failed):
Incorrect password.
```

Postoji i nekoliko osnovnih metoda zaštite od ovakvih napada. One uključuju slijedeće korake:

- ograničenje broja neuspješnih pokušaja pristupa sustavu,
- zabranu pristupa s IP adrese s koje su došli neuspjeli pokušaji pristupa,
- redovito traženje zapisa o neuspješnim pokušajima pristupa u dnevničkim zapisima,
- zatvaranje korisničkih računa za tzv. "gost" korisnike jer će oni biti prva točka upada za potencijalne napadače te
- kreiranje samo jednog korisničkog računa s najvišim ovlastima.

Uz to postoje i alati za zaštitu od *brute force* napada koji se uglavnom temelje na prethodno navedenim koracima. Njihova uloga svodi se na automatizaciju opisanih postupaka tj. na analizu dnevničkih zapisa i onemogućavanje pristupa sustavu s napadajućih IP adresa. Primjeri takvih alata koji se distribuiraju besplatno su:

- APF vatrozid & BFD skripta – rješenje temeljeno na *IPtables* vatrozidu koje nudi i tzv. *anti-dos* zaštitu i zaštitu od *brute force* napada (eng. *Brute Force Detection* - BFD). Zaštita se temelji na analizi dnevničkih zapisa (podržana su dva formata zapisa – *system kernel log* i *snort portscan log*) na osnovu kojih se poduzimaju određene akcije. Konfiguracijom je moguće odabrati slijedeće:
  - dubinu analize dnevničkog zapisa, odnosno broj redaka za analizu,
  - prag tolerancije, tj. broj neuspjelih pokušaja pristupa nakon kojih se detektira napad te
  - akcije nakon detekcije, odnosno izmjene vatrozidnih pravila kojima se onemogućava daljnji pristup poslužiteljima ili IP adresama s kojih je došao napad.
- Log Watch – alat za praćenje i analizu dnevničkih zapisa koji kreira periodičke izvještaje o aktivnosti poslužitelja. Izvještaji sadrže podatke o utrošenom diskovnom prostoru, uspjelim i neuspjelim pokušajima pristupa poslužitelju i slične informacije. Neuspjeli pokušaji se u izvještaju evidentiraju na slijedeći način:

```
anonymous/none from (IP HERE): 8 Time(s)
anonymous/password from (IP HERE): 8 Time(s)
```

```
guest/none from (IP HERE): 8 Time(s)
guest/password from (IP HERE): 8 Time(s)
root/password from (IP HERE): 24 Time(s)
```

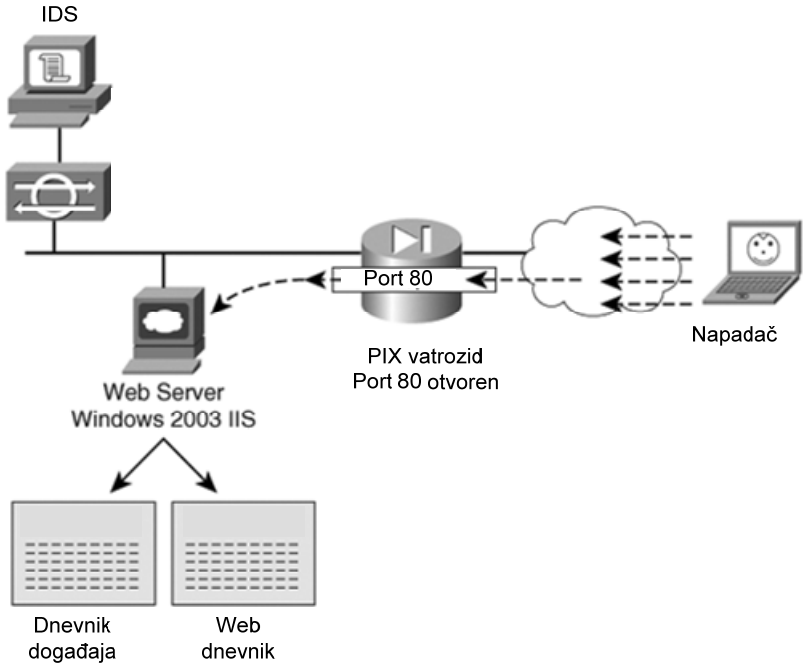
Dodatna aktivnost koja neizravno pomaže u zaštiti od *brute force* napada je prijava napadača. Naime nakon detekcije napada i provedbe zaštite blokadom izvorišne IP adrese dobro je tu adresu prijaviti davatelju usluge pristupa Internetu. Web stranica www.dnsstuff.com omogućava dobivanje podataka o davatelju usluge pristupa na osnovu IP adrese. Među podacima o davatelju usluge Internet pristupa obično se nalaze i kontakt podaci odgovorne osobe kojoj se može poslati poruka o počinjenom *brute force* napadu. U poruku je dobro uključiti dijelove dnevničkog zapisa koji evidentiraju napad te podatke o IP adresi s koje je napad došao i vremenu napada .

3.1. Detekcija i zaštita od *brute force* napada na Windows web poslužitelju

Osim izravno na poslužitelje *brute force* napadi mogu se usmjeriti i na web aplikacije. One su obično zaštićene na dva načina:

- standardnom HTTP autentikacijom - u slučaju Windows web poslužitelja ona je vezana uz SAM (eng. *Security Account Database*) bazu podataka u kojoj se nalaze podaci o korisničkim računima, te
- autentikacijom putem web forme - nešto je složenija jer zahtijeva razvijanje zasebne web stranice koja služi za autentikaciju kao i baze podataka u kojoj će se nalaziti autentikacijski podaci.

Na slijedećoj slici prikazana je standardna konfiguracije Windows poslužitelja na kojem se nalazi web aplikacija.



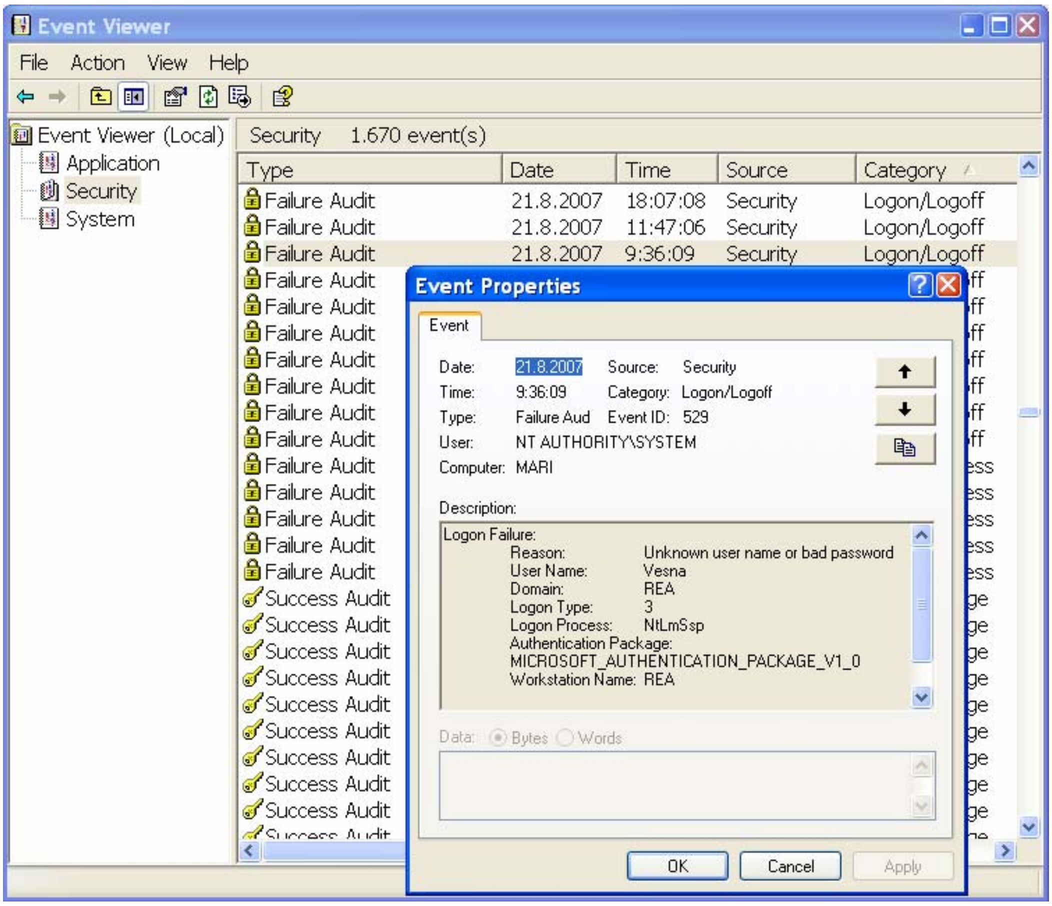
Slika 1. Standardna konfiguracija Windows poslužitelja web aplikacije

U ovoj konfiguraciji web poslužitelj je zaštićen PIX vatrozidom i IDS sustavom za detekciju uljeza (IDS eng. *Intrusion Detection System* - IDS), ali je *brute force* napad unatoč tome moguće izvesti. Testiranja potvrđuju da PIX vatrozid i IDS sustav ne detektiraju takav napad već se tragovi o napadu moraju potražiti u dnevničkim zapisima samog web poslužitelja.

Na Windows web poslužitelju postoje dvije vrste dnevničkih zapisa:

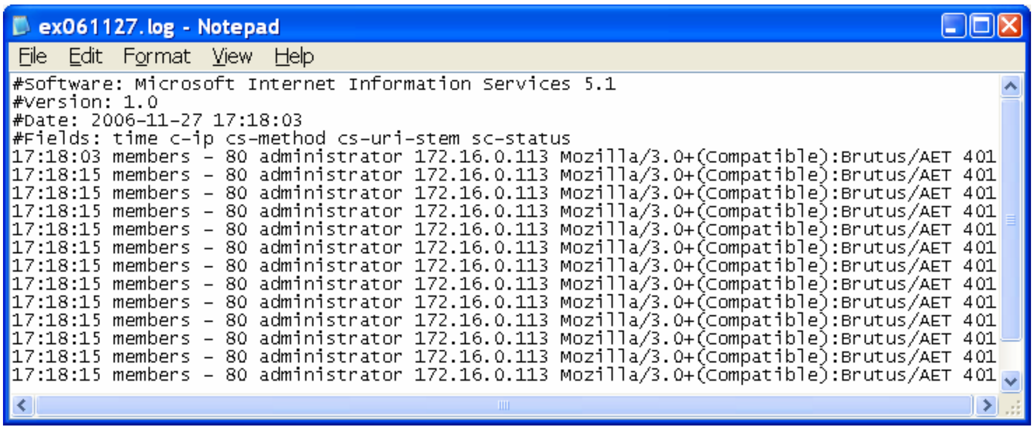
- dnevnički zapis sigurnosnih događaja (eng. *Windows Security Event Log*) i
- dnevnički zapis IIS poslužitelja (uobičajeno se nalazi na lokaciji C:\windows\system32\logfiles\w3svc1).

U slučaju *brute force* napada dnevnički zapis sigurnosnih događaja će sadržavati zapise o neuspješnim pokušajima pristupa web poslužitelju označenih s identifikatorom „Event ID 529"kao što je prikazano na slijedećoj slici.



Slika 2. Dnevnički zapis sigurnosnih događaja na Windows web poslužitelju

Dnevnički zapis IIS poslužitelja će također pokazivati evidenciju neuspješnih pokušaja pristupa u ovom slučaju označenih kao greške tipa 401 kao što je prikazano na slijedećoj slici.



ex061127.log - Notepad

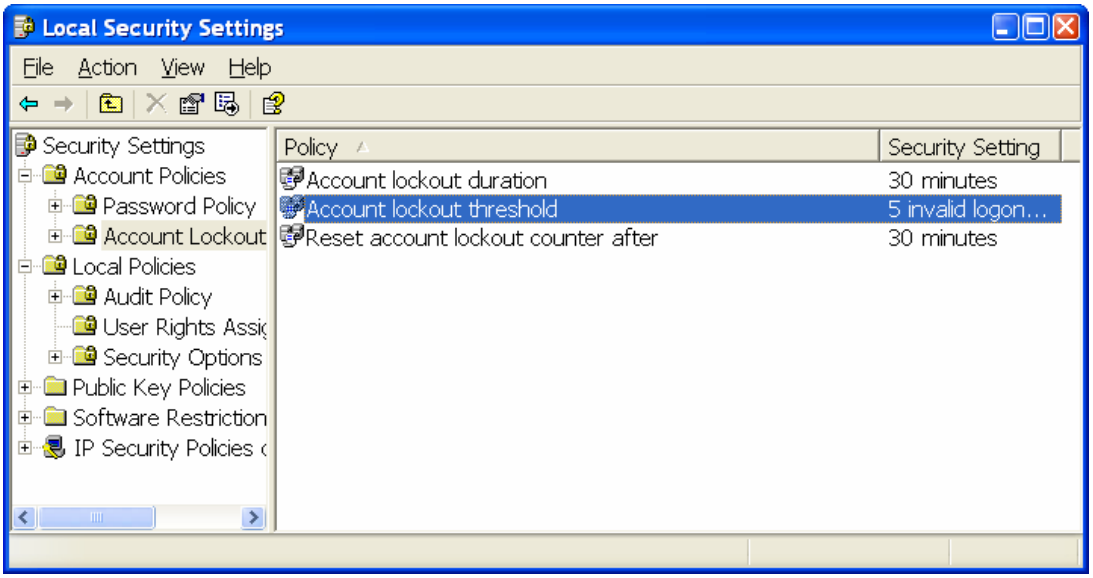
File Edit Format View Help

```
#Software: Microsoft Internet Information Services 5.1
#Version: 1.0
#Date: 2006-11-27 17:18:03
#Fields: time c-ip cs-method cs-uri-stem sc-status
17:18:03 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
17:18:15 members - 80 administrator 172.16.0.113 Mozilla/3.0+(Compatible):Brutus/AET 401
```

Slika 3. Dnevnički zapis IIS web poslužitelja

Kod analize neuspješnih pokušaja prijave ne treba biti previše paranoičan. Uobičajeno je u dnevnicima pronaći nekoliko neuspješnih pokušaja pristupa, ali tek ako ih se u zapisima nalazi nekoliko desetaka za redom, može se govoriti o *brute force* napadu.

Zaštitu od ovakvih napada u slučaju standardne HTTP autentikacije moguće je ostvariti ograničenjem maksimalnog broja neuspješnih pokušaja pristupa aplikaciji. Postavke *Administrative Tools\Local Security Policy* prikazane na slijedećoj slici.



Slika 4. Postavke ograničenja maksimalnog broja neuspješnih pokušaja pristupa

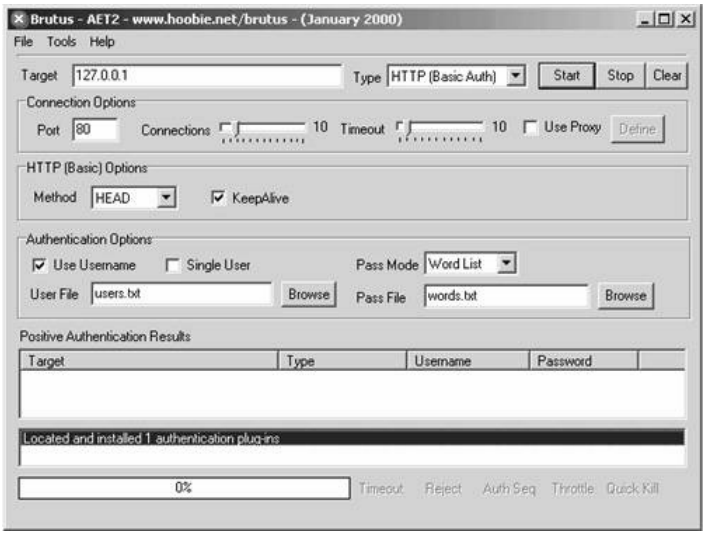
U slučaju autentikacije putem web forme za ograničenje maksimalnog broja neuspješnih pokušaja pristupa bilo bi potrebno mijenjati izvoran kod web aplikacije, što je već mnogo opsežniji zadatak. Osim toga, ovakav pristup ima i moguću nuspojavu – u slučaju napada korištenjem nekog postojećeg korisničkog imena nedužnom korisniku će automatska zaštita onemogućiti pristup web aplikaciji. U takvoj situaciji takvim korisnicima bi trebalo otvoriti nove korisničke račune što je dodatan posao za administratore.

Drugi način zaštite je već ranije opisana zaštita temeljena na analizi dnevničkih zapisa i blokiranju određenih IP adresa na razini vatrozida. Međutim, ako se napadač nalazi iza NAT (eng. *Network Addess Translation*) uređaja ili posrednog (eng. *Proxy*) poslužitelja blokiranje njegove IP adrese će za posljedicu imati i blokiranje pristupa svim korisnicima koji se nalaze iza tog NAT uređaja ili posrednog poslužitelja.

4. Testiranje ranjivosti na *brute force* napade

Kako bi se sustav efikasno zaštitio od *brute force* napada, potrebno ga je dobro testirati i tako dobiti uvid u njegovo ponašanje u kritičnim okolnostima. Na osnovu dobivene slike potrebno je predložiti najbolje mjere zaštite. Za testiranje je najbolje koristiti alate koji se inače mogu koristiti i za napade. Jedan od takvih je *Brutus*.

On je jednostavan i prilično moćan alat za provedbu *brute force* napada. Besplatan je i dostupan samo u inačici za Microsoft Windows operacijski sustav na slijedećoj lokaciji: http://www.hoobie.net/brutus/. *Brutus* alatom je moguće izvesti napad od 30 000 pokušaja u minuti prema standardnoj HTTP autentikaciji, autentikaciji putem web forme ili prema FTP, POP3 ili Telnet pristupu. Sučelje alata prikazano je na slijedećoj slici.



Slika 5. Sučelje *Brutus* alata

Kao što je vidljivo sučelje je prilično jednostavno, a korisniku omogućava prilagodbu postavki broja veza i vremenskog razmaka između dva uzastopna pokušaja pristupa. Budući da se zaštite mnogih web stranica temelje na broju veza ili broju uzastopnih pokušaja s iste IP adrese, izmjenom postavki takva se zaštita ponekad može zaobići.

5. Zaključak

Kao što je vidljivo iz dokumenta *brute force* napad ne zahtijeva veliku znanje ni umijeće napadača, što samo znači da je krug potencijalnih napadača vrlo širok. Zaštita od ovih napada je moguća, ali samo u ograničenoj mjeri sredstvima koja stoje na raspolaganju administratorima sustava. Glavnina odgovornosti ipak ostaje na samim korisnicima i količini truda koju oni žele uložiti u zaštitu svojih zaporki da bi ih učinili težim za probijanje. Ono što administratori mogu učiniti u tom pogledu je definirati sigurnosne politike i preporuke kojima će korisnicima ukazati na tu opasnost i prisiliti ih na korištenje kompleksnijih zaporki. Uz metode zaštite opisane u ovom dokumentu i savjesne korisnike koji se pridržavaju preporuka, rizik od uspjeha *brute force* napada je vrlo malen.

6. Reference

[1] Zaštita od *brute force* napada , http://www.webhostgear.com/240.html, travanj 2005.
[2] Kriptografski *brute force* napadi, http://www.cl.cam.ac.uk/~rnc1/brute.html, listopad 2001.
[3] *Brute force* testiranje ranjivosti, http://www.resultspk.net/penetration_testing_and_network_defense/ch07lev1sec6.html, rujan 2005.
[4] Proboj DES algoritma, http://en.wikipedia.org/wiki/EFF_DES_cracker, srpanj 2007.
[5] Općenito o *brute force* napadu, http://en.wikipedia.org/wiki/Brute-force_search, kolovoz 2007.
[6] Općenito o *brute force* napadu, http://en.wikipedia.org/wiki/Brute_force_attack, kolovoz 2007.