HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Analiza YAPH programskog paketa

CCERT-PUBDOC-2003-02-03

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

Primarna svrha *proxy* poslužitelja je privremeno pohranjivanje podataka dohvaćenih s Interneta (tzv. *caching*) i njihovo posluživanje klijentima na lokalnoj mreži. Budući da je propusnost mrežnog linka kojim je lokalna mreža spojena na Internet redovito mnogo manja od propusnosti same lokalne mreže, *proxy* poslužiteljima se želi postići rasterećenje glavnog voda i prividno ubrzanje prometa prema klijentima. Može se reći da se *proxy* poslužitelj ponaša kao posrednik između klijenta i poslužitelja.
Propusti u konfiguraciji poslužitelja vrlo lako mogu rezultirati otvorenim *proxy* poslužiteljima. Pod otvorenim *proxy* poslužiteljem smatra se onaj poslužitelj koji dozvoljava spajanja računalima koja se ne nalaze u njegovoj LAN mreži (dozvoljava spajanje svim računalima na Internetu) tj. dozvoljava i spajanje neovlaštenim korisnicima. Otvorene *proxy* poslužitelje neovlašteni korisnici koriste za prikrivanje identiteta i razne neovlaštene aktivnosti, te ih je zbog toga potrebno detektirati i onemogućiti. `Yaph` je program koji služi pronalaženju takvih *proxy* poslužitelja u svrhu njihovog lakšeg uklanjanja.

2. Instalacija i konfiguracija

`Yaph` se distribuira u `tar.gz` paketu koji se otpakirava naredbom:

```
tar -xzvf yaph.tar.gz
```

Najnovija inačica ovog programa može se pronaći na adresi http://proxylabs.netwu.com/yaph/download.html. Instalacija započinje pokretanjem `./configure` skripte, koja provjerava konfiguraciju sustava na kojem se `Yaph` instalira. Program se prevodi naredbom `make`, a instalira naredbom `make install`. Potrebno je napomenuti da su za pokretanje `make install` naredbe potrebne administratorske ovlasti.
Za svoj rad `Yaph` koristi `nmap` (http://www.insecure.org/nmap) i `Proxy Chains` (http://proxychains.sourceforge.net/) alate. Ukoliko se u potpunosti žele iskoristiti mogućnosti ovog alata svakako je potrebno instalirati `nmap` i `Proxy Chains`.
Prilikom instalacije, u `/etc` direktorij smješta se datoteka `yaph.conf` u kojoj se nalaze parametri kojima se kontrolira izvođenje programa. Osim `yaph.conf` datoteke u `/etc` direktoriju, moguće je kreirati i vlastitu datoteku i smjestiti ju u direktorij iz kojega se program pokreće. Prilikom pokretanja, program će prvo potražiti konfiguracijsku datoteku u direktoriju iz kojega se pokreće, a ukoliko ju nije pronašao iskoristiti će datoteku u `/etc/` direktoriju.
Valjane opcije u `yaph.conf` datoteci su sljedeće:

- **MaxCheckThreads** – ova opcija ograničava broj paralelnih podprocesa koji su pokrenuti. Velik broj podprocesa, zbog ograničenih hardverskih resursa, može rezultirati netočnim provjerama.
- **TcpReadTimeOut** – određuje vrijeme čekanja u milisekundama za funkciju `tcp_read()`.
- **TcpConnectTimeOut** – određuje vrijeme čekanja u milisekundama za funkciju `tcp_connect()`.
- **ContentHost** – FQDN ime ili IP adresa računala koje će proxyfind koristiti za provjeru ispravnosti tunela.
- **ContentPort** – određuje port na kojem se nalazi poslužitelj s sadržajima.
- **ContentRequest** – ova opcija predstavlja znakovni niz koji će se poslati ContentHost računalu nakon uspostave tunela preko otvorenog *proxy* poslužitelja. Ukoliko se na ovo mjesto upiše krivi niz, provjera ispravnosti uspostavljenog tunela neće biti moguća.
- **ContentData** – predstavlja znakovni niz koji se očekuje u odgovoru *ContentHost* računala. Pogrešno upisana vrijednost u ovom parametru može spriječiti provjeru ispravnosti.
- **ResultFile** – označava ime datoteke u koju se spremaju rezultati pretraživanja. Podaci su zapisani u proxychains formatu (tip *proxy* poslužitelja, ime računala, port). Ukoliko se umjesto imena datoteke upiše vrijednost STDOUT, rezultati će se ispisati na standardni izlaz (konzola).

- **LogFile** – označava datoteku u koju se spremaju interne poruke koje program generira. Ova vrijednost je inicijalno postavljena na standardni izlaz.
- **LogLevel** – ovaj parametar određuje koliko će opsežan biti broj log poruka. Dozvoljeni su brojevi od 0 do 4 (što je veći broj, prijavljuje se više grešaka).

3. Mogućnosti programa

`Yaph` je u mogućnosti pronaći i prepoznati otvorene SOCKS v4, v5 i HTTP *proxy* poslužitelje. Kada program pronađe *proxy* poslužitelj, pokušava preko njega tunelirati svoje upite na neki poslužitelj (inicijalno www.yahoo.com) i ukoliko dobije valjani odgovor, *proxy* poslužitelj se proglašava otvorenim. HTTP *proxy* poslužitelji provjeravaju se isključivo CONNECT metodom, tj. kroz njih se ne pokušava tuneliranje. Načelno, ovom programu se mora zadati lista *proxy* poslužitelja koji će se provjeravati, ali u kombinaciji sa `nmap` i `proxychains` alatima, `Yaph` je u mogućnosti izvoditi vrlo komplicirana pregledavanja mreže, čak i u pritajenom načinu rada.

3.1. Provjera liste *proxy* poslužitelja

Ukoliko mu se zada lista *proxy* poslužitelja, `Yaph` će ispitati da li su poslužitelji otvorenog tipa. Podržani formati liste *proxy* poslužitelja su `Proxy Hunter` format i `Proxy Chains` format:

- Proxy Hunter:

```
192.168.1.2:8080@HTTP
  192.168.1.3:8080@SOCKS4
  192.168.1.4:8080@SOCKS5
```

- Proxy Chains:

```
http 192.168.1.2 8080
  socks4    192.168.1.3 8080
  socks5    192.168.1.4 8080
```

Primjer:
Sljedeća linija provjeriti će poslužitelje navedene u datoteci `proxy_list.txt` u `Proxy Hunter` formatu.

```
cat proxy_list.txt | sort | uniq | proxychains yaph --
use_hunter_stdin
```

Za provjeru će se koristiti *proxy* poslužitelj naveden u `proxychains.conf` datoteci. Naredbe `sort` i `uniq` koriste se za uređivanje liste i uklanjanje višestrukih zapisa.
Za provjeru *proxy* poslužitelja moguće je koristiti i interaktivni način rada u kojemu korisnik u konzoli upisuje poslužitelje koji će se provjeravati.

```
# yaph -use_chains_stdin
[Thu 06/Feb 09:57:29]:PID=2882:main.c:main:54:  START
http 192.168.1.145 80
PID=2884:threads.c:file_parser_thread:424:  parsed: 192.168.1.145 80
0
PID=2882:content_utils.c:get_target:45:  Got target at
192.168.1.145:80 tcp
PID=2882:main.c:init_check:30:  Targets in progress = 1 ..
PID=2882:main.c:init_check:30:  Targets in progress = 2 ..
PID=2885:threads.c:check_http:243:  http-connect proxy test STARTED
against 192.168.1.145:80 tcp
PID=2885:threads.c:check_http:268:  Trying to set up http tunnel via
192.168.1.145:80 tcp. HTTP-CONNECT command sent.
PID=2885:threads.c:check_http:279:  Server denied to set up http
tunnel : HTTP/1.1 405 Method Not Allowed
PID=2885:threads.c:check_http:302:  http-connect proxy test FINISHED
against 192.168.1.145:80 tcp
```

Umjesto parametara `http` ili `socks` u `ProxyChains` formatu, moguće je navesti parametar `any` kao tip poslužitelja. U tom slučaju poslužitelj će se provjeravati za sva tri tipa (http, socks4 i socks5).

3.2. Pregledavanje mreže

U kombinaciji sa `nmap` programom za pregledavanje otvorenih portova, `Yaph` se može koristiti za pregledavanje mreže u svrhu pronalaženja otvorenih `proxy` poslužitelja. Sve opcije `nmap`-a podržane su transparentno, osim `-oG` opcije za određivanje log datoteke koju interno koristi `Yaph`. Zbog brojnih mogućnosti `nmap`-a moguće je obavljati vrlo kompleksna pregledavanja mreže.
Primjer:
Sljedeći primjer prikazuje upotrebu `nmap`-a u kombinaciji s `Yaph`-om:

```
# yaph --use_nmap -sS  -p 1080,8080,3128 192.168.1*
[Thu 06/Feb 14:05:38]:PID=6148:threads.c:nmap_parser_thread:332:
TCP probe port is 4665

[Thu 06/Feb 14:05:38]:PID=6145:main.c:main:54:  START
[Thu 06/Feb 14:05:42]:PID=6148:threads.c:nmap_parser_thread:332:  #
nmap (V. 3.00) scan initiated Thu Feb  6 14:05:38 2003 as: nmap -n -
oG - -randomize_hosts -PT4665 -sS -p 1080,8080,3128 192.168.1*

[Thu 06/Feb 14:05:42]:PID=6148:threads.c:nmap_parser_thread:332:
Host: 192.168.154 () Status: Up

[Thu 06/Feb 14:05:42]:PID=6148:threads.c:nmap_parser_thread:332:
Host: 192.168.1.117 ()        Ports: 3128/open/tcp//squid-
http///Ignored State: closed (2)

[Thu 06/Feb 14:05:42]:PID=6145:content_utils.c:get_target:45:  Got
target at 192.168.1.117:3128 tcp
[Thu 06/Feb 14:05:42]:PID=6145:main.c:init_check:30:  Targets in
progress = 1 ..
[Thu 06/Feb 14:05:42]:PID=6148:threads.c:nmap_parser_thread:332:
Host: 192.168.1.103 ()        Status: Up

[Thu 06/Feb 14:05:42]:PID=6149:threads.c:check_http:243:  http-
connect proxy test STARTED against 192.168.1.117:3128 tcp
[Thu 06/Feb 14:05:42]:PID=6145:main.c:init_check:30:  Targets in
progress = 2 ..
[Thu 06/Feb 14:05:42]:PID=6149:threads.c:check_http:268:  Trying to
set up http tunnel via 192.168.1.117:3128 tcp. HTTP-CONNECT command
sent.
[Thu 06/Feb 14:05:42]:PID=6145:main.c:init_check:30:  Targets in
progress = 3 ..
[Thu 06/Feb 14:05:42]:PID=6150:threads.c:check_socks4:26:  Socks4
proxy test STARTED against 192.168.1.117:3128 tcp
[Thu 06/Feb 14:05:42]:PID=6150:threads.c:check_socks4:50:  Trying to
set up  tunnel via 192.168.1.117:3128 tcp, Socks4 command sent
[Thu 06/Feb 14:05:42]:PID=6151:threads.c:check_socks5:106:  Socks5
proxy test STARTED against 192.168.1.117:3128 tcp
[Thu 06/Feb 14:05:42]:PID=6151:threads.c:check_socks5:128:  Trying
to set up  tunnel via 192.168.1.117:3128 tcp, Socks5 'method'
command sent
[Thu 06/Feb 14:05:42]:PID=6149:threads.c:check_http:279:  Server
denied to set up http tunnel : HTTP/1.0 403 Forbidden

[Thu 06/Feb 14:05:42]:PID=6149:threads.c:check_http:302:  http-
connect proxy test FINISHED against 192.168.1.117:3128 tcp
```

```
[Thu 06/Feb 14:05:56]:PID=6145:main.c:main:87:  Targets in progress
= 0 ..
[Thu 06/Feb 14:05:56]:PID=6145:main.c:main:90:  DONE
```

Yaph je pokrenut tako sa pregledava cijelu mrežu 192.168.1.0/24 tražeći *proxy* poslužitelje na portovima 1080, 8080 i 3128. Zbog dužine ispisa prikazane su samo one linije koje se odnose na pregledavanje računala 192.168.1.117.
Pregledavanje mreže smatra se neovlaštenom djelatnošću, tako da se izvođenje ovakvih radnji preporučuje samo na vlastitoj mreži.

3.3. Pritajeno pregledavanje mreže

Yaph posjeduje i mogućnost pritajenog pregledavanja mreže pomoću Proxy Cahins alata. Proxy Chains nudi mogućnost tuneliranja veze kroz neki *proxy* poslužitelj u svrhu prikrivanja vlastite IP adrese. Prilikom pregledavanja računala, u njihovim log datotekama biti će zapisane adrese *proxy* poslužitelja preko kojega je vršeno pregledavanje. Ako se tuneliranje kroz *proxy* poslužitelj koristi u kombinaciji sa nmap opcijom –D (*decoy*) koja omogućuje lažiranje IP adrese sa koje se provodi pregledavanje, dobiva se vrlo efektan mehanizam za skrivanje izvorne IP adrese sa koje se pokreće Yaph. Sintaksa naredbe koja bi koristila pritajeno pregledavanje u kombinaciji sa lažiranjem izvorne IP adrese izgleda ovako:

```
proxychains yaph –D92.180.37.2,193.45.36.10,203.18.46.95
192.168.35.*
```

Adresa poslužitelja kroz kojega će se tunelirati veza zapisana je u proxychains.conf datoteci.

4. Zaključak

Yaph je program namijenjen pretežno administratorima, koji pomoću njega vrlo lako mogu uočiti otvorene *proxy* poslužitelje na vlastitoj mreži. Iako program posjeduje mogućnosti pritajenog pregledavanja potrebno je napomenuti da se pregledavanje tuđih mreža u potrazi za otvorenim *proxy* poslužitelja smatra zloupotrebom.
Za bolje upoznavanje sa svim opcijama koje ovaj alat posjeduje i mogućnostima njihove primjene preporučuje se i detaljno proučavanje dokumentacije nmap i Proxy Chains alata.