

Re: pf rdr + netsted : reinject loop...

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/questions/2007-08/msg02203.html>

- *From:* Mel <fbbsd.questions@xxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Fri, 31 Aug 2007 22:09:42 +0200
-

On Friday 31 August 2007 18:27:26 Norberto Meijome wrote:

On Fri, 31 Aug 2007 17:40:06 +0200

Mel <fbbsd.questions@xxxxxxxxxxxxxxxxxxxxxxxx> wrote:

netsted's output is (part) :

```
Script started on Fri Aug 31 07:52:12 2007
[root@localhost /usr/home/luser]# netsted tcp 10101 0 0
s/FOO/BAR
netsted 0.01b by Michal Zalewski <lcamtuf@xxxxxx>
[*] Parsing rule s/FOO/BAR ...
[+] Loaded 1 rules...
[+] Listening on port 10101/tcp.
[+] Using dynamic (transparent proxy) forwarding.

[+] Got incoming connection from 172.16.82.81:1178 to
127.0.0.1:10101
[*] Forwarding connection to 127.0.0.1:10101
[+] Got incoming connection from 127.0.0.1:51337 to
127.0.0.1:10101
[*] Forwarding connection to 127.0.0.1:10101
[+] Caught client -> server packet.
```

I think you need to figure out what this 'transparent proxy mode' of netsted does, cause it should under no circumstances forward to itself...

it simply forwards the packet to the dst_ip:dst_port it originally had. But, as Daniel H pointed out, those packets had been rewritten by pf's rdr to go TO netsted's ip:port hence netsted wont change anything. It works fine in non-proxy mode, but as I said in my first msg, that is not an option for me.

So the obvious question is how to get the packets to netsted's IP:PORT without having the packet's original destination IP/PORT changed....maybe incorporating the netsted code into a socks5-compatible server (in my case,

Re: pf rdr + netseed : reinject loop...

the app that generates the packets understands SOCKS). Alas, I am drawing a blank here atm.

Otherwise, i can only think that a new netgraph node would perform better than my current pf + netseed approach....

Figured I'd take a shot at it and it works:

```
# ./netseed tcp 10101 0 0 s/boo/GET/
netseed 0.01b by Michal Zalewski <lcamtuf@xxxxxx>
[*] Parsing rule s/boo/GET/...
[+] Loaded 1 rules...
[+] Listening on port 10101/tcp.
[+] Using dynamic (transparent proxy) forwarding.
[+] Got incoming connection from 11.22.33.44:27712 to 127.0.0.1:10101
[*] Forwarding connection to 55.66.77.88:80
[+] Caught client -> server packet.
```

Renamed the ip's to protect the innocent, but that's all. I typed boo / HTTP/1.0 and got back a solid page of html. Patch inlined below sig. I'm surprised no one ever caught up on this, seeing the makefile is last modified in 2005 :)

--

Mel

```
--- orig/netseed.c 2007-08-31 21:51:51.000000000 +0200
+++ work/netseed.c 2007-08-31 21:51:31.000000000 +0200
@@ -11,6 +11,12 @@
#include <ctype.h>
#include <stdlib.h>
#include <signal.h>
+#ifdef USE_PF
+#include <sys/ioctl.h>
+#include <net/if.h>
+#include <net/pfvar.h>
+#include <sysexits.h>
+#endif

#define VERSION "0.01b"
#define MAXRULES 50
@@ -254,11 +260,19 @@
signal(SIGCHLD,sig_chld);

// Am I bad coder?;>
+ /* Yeah, comments should be useful and frequent and not in C++ format. */

while (1) {
struct sockaddr_in s;
int x,l=sizeof(struct sockaddr_in);
int conho,conpo;
```

Re: pf rdr + netseed : reinject loop...

Re: pf rdr + netsed : reinject loop...

```
+ #ifdef USE_PF
+ struct pfio natlook;
+ int fd;
+ socklen_t clen; /* client length */
+ struct sockaddr_in *client; /* client socket */
+ #endif
+
+ usleep(1000); // Do not wanna select ;P
+ if ((csock=accept(lsock,(struct sockaddr*)&s,&l))>=0) {
+ fcntl(csock,F_SETFL,O_NONBLOCK);
+ @@ -266,8 +280,51 @@
+ l=sizeof(struct sockaddr_in);
+ getsockname(csock,(struct sockaddr*)&s,&l);
+ printf(" to %s:%d\n", inet_ntoa(s.sin_addr), ntohs(s.sin_port));
+ /* The logic here is that it receives an unmodified dest address,
+ * however that's not the case with pf. */
+ #ifdef USE_PF
+ /* We also need the client peer to look up the nat in pf, blatantly
+ * borrowed from ftp-proxy(8). */
+ clen = sizeof(struct sockaddr_in);
+ client = (struct sockaddr_in *)malloc(clen);
+ getpeername(csock, (struct sockaddr *)client, &clen);
+ memset((void *)&natlook, 0, sizeof(natlook));
+ natlook.af = AF_INET;
+ natlook.saddr.addr32[0] = client->sin_addr.s_addr;
+ natlook.daddr.addr32[0] = s.sin_addr.s_addr;
+ natlook.proto = IPPROTO_TCP;
+ natlook.sport = client->sin_port;
+ natlook.dport = s.sin_port;
+ /* NOTE: It works with PF_OUT, even though rdr rule is on incoming
+ * traffic in my tests. More research into natlook.direction is needed
+ * here.
+ */
+ natlook.direction = PF_OUT;
+ /*
+ * Open the pf device and lookup the mapping pair to find
+ * the original address we were supposed to connect to.
+ */
+ fd = open("/dev/pf", O_RDWR);
+ if (fd == -1) {
+ printf("No permission to open /dev/pf, see ya\n");
+ exit(EX_UNAVAILABLE);
+ }
+
+ if (ioctl(fd, DIOC NATLOOK, &natlook) == -1) {
+ printf(
+ "pf nat lookup failed %s:%hu\n",
+ inet_ntoa(client->sin_addr),
+ ntohs(client->sin_port));
+ close(fd);
+ exit(EX_UNAVAILABLE);
+ }
+ }
+ }
```

Re: pf rdr + netsed : reinject loop...

Re: pf rdr + netseed : reinject loop...

```
+ }
+ close(fd);
+ conpo=ntohs(natlook.rdport);
+ conho=natlook.rdaddr.addr32[0];
+ #else
conpo=ntohs(s.sin_port);
conho=s.sin_addr.s_addr;
+ #endif
if (fixedport) conpo=fixedport;
if (fixedhost) conho=fixedhost;
s.sin_addr.s_addr=conho;
--- orig/Makefile 2001-01-05 02:46:32.000000000 +0100
+++ work/Makefile 2007-08-31 21:35:32.000000000 +0200
@@ -1,4 +1,4 @@
-CFLAGS = -Wall -fomit-frame-pointer -O9
+CFLAGS = -Wall -fomit-frame-pointer -O9 -DUSE_PF

all: netseed
```

freebsd-questions@xxxxxxxxxxx mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-questions>

To unsubscribe, send any mail to "freebsd-questions-unsubscribe@xxxxxxxxxxx"