Network Exploitation with Ncrack

ithilgore
sock-raw.org
# whoami

- Network security researcher (sock-raw.org)
- Exploiting TCP and the Persist Timer Infiniteness (Phrack #66)
- Abusing Network Protocols (stealthy portscanning through XMPP exploitation)
- Nmap/Ncrack development

Contact:
ithilgore@sock-raw.org
ithilgore.ryu.1@gmail.com
twitter.com/ithilgore

http://sock-raw.org/gpgkey
How it all started

It was a bug. Not a feature.

First significant feedback to Nmap project.

http://seclists.org/nmap-dev/2008/q4/543
ip->ip_len != IP length

/usr/src/linux-2.6.26/net/ipv4/raw.c

iphlen = iph->ihl * 4;
if (iphlen >= sizeof(*iph) && iphlen <= length) {
    if (!iph->saddr)
        iph->saddr = rt->rt_src;
    iph->check = 0;
    // iph->tot_len = htons(length);
    if (!iph->id)
        ip_select_ident(iph, &rt->u.dst, NULL);
...

No shady business there, sir.

/usr/src/sys/kern/raw_ip.c

if (((ip->ip_hl != (sizeof (*ip) >> 2)) && inp->inp_options) || (ip->ip_len > m->m_pkthdr.len) || (ip->ip_len < (ip->ip_hl << 2))) {
    INP_UNLOCK(inp);
    m_freem(m);
    return EINVAL;
}

Linux being too strict. No problem: recompile kernel
150-175 Open Source Organizations

3-4 months

1000 students

4,500k – 5000k $ stipends

~26k lines of code (Ncrack)
The goal: Ncrack

Ncrack is designed to be a fast and flexible network authentication cracker. You can point it at a service (ssh, msrpc, http, imap, pop3, SNMP, telnet, ftp, etc.) and it will make repeated authentication attempts. The goal is, of course, to find working credentials by brute force. It is a very handy tool to have during pen-tests, as many/most users still choose weak passwords.

http://seclists.org/nmap-dev/2009/q2/238
RFC on Ncrack, A new network authentication cracker
Why?

- Weak passwords more common than exploits
- Brute force scripts most popular in NSE
- Competitors (*THC-Hydra, Medusa* etc)
  - not very actively maintained
  - some are way old and buggy (*Brutus, TSGrinder*)
  - portability problems (esp. Windows)
  - limitations (multiple hosts, timing fine-graining)
- Top 15 security tools ([sectools.org](http://sectools.org)) are cracking natured
Architecture

- < handles connection & authentication endings >
- < timing & dynamic adaptation >
- < registers Nsock callback handlers >
- < checks network conditions >
- < calls protocol modules >

Ncrack Core Engine 0.4 alpha
Modules

nsock_loop

select(2)

SSH module

State 0
nsock_read/write

State 1
nsock_read/write

State N
ncrack_module_end

state machine

Ncrack Engine
Nsock above TCP => no SOCK_RAW

Problem: timing algo without power over packets

Solution: rely on RST, timeouts and statistics
many parallel probes

normal replies

many parallel probes

target
even more parallel probes

normal replies

target
even more parallel probes

normal replies & RSTs

target
decrease maximum probes

normal replies & RSTs

decrease maximum probes

target
keep decreasing maximum probes

normal replies & RSTs

target
System Balanced

start increasing probes again

normal replies only

target
ncrack probes

mean = ideal parallelism?
What about timeouts?

Much more difficult to handle:
– might be due to network failure
– may stem from firewall rulesets
– could be combined with RSTs
or
– may result from accidentally DoS-ing the scanned service
In reality, our metric is not the amount of RSTs or timeouts but the authentication rate.

Ideally: use a trial-and-error approach and save a history of different performances.
Timing algorithm

Experimentation phase:

1. keep increasing parallel probes until:
   a. authentication rate drops OR
   b. authentication rate stays the same OR
   c. any error occurs (RST, timeout)
2. drop limit of probes if one of the above happens
3. Goto 1 until you have an adequate sample
Chicken and egg problem

How do we know we reached the ideal parallelism?

*Answer:* We don’t. We always have to rely on past samples, which have been gathered through trial-and-error.
In search of the 
Golden Ratio

- Accuracy
- Speed
- Resource saving

**Problem**: Network conditions are dynamic and often random.

Temporarily use the mean of the samples and rerun sample-gathering algorithm at intervals.
Time fine-graining

User defined options which override Ncrack’s dynamically found values.

Timing Template (Nmap style)

-T paranoid|sneaky|polite|normal|aggressive|insane

OR

T0 - T5

possible DoS
Imposing limits

- `cl` (min connection limit): minimum number of concurrent parallel connections

VS

- `CL` (max connection limit): maximum number

- `cd` (connection delay): adjust delay time between each new connection

esp. useful for resource saving

- `at`: authentication attempts per connection
Punching the firewall hole

Scenario: Crack at least one SSH account of host “diogenis.ceid.upatras.gr” listening on port 45120 without alerting/triggering any firewall/IDS.

Assumption: Blocks IP if connections > 2 per minute
**sshd_config defaults**

MaxAuthTries: 6  
MaxStartups: 10

Our attack will take place during the nights only (use -to and cron)

Ncrack initially sends a *reconnaissance probe* to figure out maximum authentication attempts per connection

maximum attempts per connection (use -at)

maximum concurrent connections per IP (use –CL)
$ time ncrack \
> ssh://diogenis.ceid.upatras.gr:45120,CL=1,at=10,cd=1m \n> --passwords-first -d6

Starting Ncrack 0.4ALPHA ( http://ncrack.org ) at 2011-05-06
  02:27 EEST
ssh://150.140.141.181:22 (EID 1) Connection closed by peer
ssh://150.140.141.181:22 (EID 1) Attempts: total 6 completed 6
  supported 6 --- rate 0.43
caught SIGINT signal, cleaning up
Saved current session state at:
   /home/ithilgore/.ncrack/restore.2011-05-06_02-28

real    0m16.049s
user    0m0.010s
sys     0m0.010s

1 connection only

maximum attempts per connection

time for one connection
cracking time
1 connection

cd=15s
(rest)

Goal: <= 2 connections per minute

We assumed ~15 secs per connection
$ ncrack \\
> ssh://diogenis.ceid.upatras.gr:45120,CL=1,at=6,\\
> cd=15s,to=6h -v -f --user ‘xantzis’ \\
> -P ~/lists/greeklish_pass.txt --save ~/ssh_session

keep cracking for 6 hours

delay between each new connection

quit cracking after 1 found credential

save current session to be resumed later

$ crontab -l
00 21 * * * /usr/local/bin/ncrack --resume
/home/ithilgore/ssh_session
Ncrack SSH library:
- based on OpenSSH code
- hacked socket code and substituted with Ncrack callbacks
- backwards compatibility with obscure ssh servers
- extensible for many types of authentication
Effective SSH cracking

Username list: guest, root
Password list: 12345, test, foo, bar

Default order: guest/12345, root/12345, guest/test, root/test, guest/foo, root/foo, guest/bar, root/bar
(--passwords-first to reverse order)

Problem: SSH doesn’t allow changing a username in the same connection
Use reconnaissance probe to learn the maximum authentication attempts per connection (suppose 3).

*Username list:* guest, root  
*Password list:* 12345, test, foo, bar, changeme, lala, keke, 000

Suppose 4 parallel connections:  
#1 -> guest/12345 and ‘test’ and ‘foo’  
#2 -> root/12345 and ‘test’ and ‘foo’  
#3 -> guest/bar and ‘changme’ and ‘lala’  
#4 -> root/bar and ‘changme’ and ‘lala’
**Remember**: sometimes services purposefully insert delay (2-3 sec or more) between each auth attempt.

In that case: may be better to open many connections with 1 auth attempt each and immediately close, less time than imposed delay.
Remote Desktop: the 1+ man-month task

Unique in cracking:
- tsgrinder broken
- rdesktop patches don’t really do any real work

bitmap compression =>
don’t flip that bit!
RDP Hell

Connection Initiation

Basic Settings Exchange

Channel Connection

RDP Security Commencement

Secure Settings Exchange

Licensing

Capabilities Exchange

Connection Finalization

Client

- X.224 Connection Request PDU
- X.224 Connection Confirm PDU
- MCS Connect Initial PDU with GCC Conference Create Request
- MCS Connect Response PDU with GCC Conference Create Response
- MCS Erect Domain Request PDU
- MCS Attach User Request PDU
- MCS Attach User Confirm PDU
- MCS Channel Join Request PDU(s)
- MCS Channel Join Confirm PDU(s)
- Security Exchange PDU
- Client Info PDU
- License Error PDU - Valid Client
- Demand Active PDU
- Confirm Active PDU
- Synchronize PDU
- Control PDU - Cooperate
- Control PDU - Request Control
- Persistent Key List PDU(s)
  - Font List PDU
  - Synchronize PDU
  - Control PDU - Cooperate
  - Control PDU - Granted Control
  - Font Map PDU

Server
info->memblt.opcode == 0x0cc &&
info->memblt.x == 740 &&
info->memblt.y == 448 &&
info->memblt.cx == 60 &&
info->memblt.cy == 56 &&
info->memblt.cache_id == 2

magic RDP fingerprint for Windows Vista/7/Server 2003/2008
Ncrack features pentesters will adore

- Target input straight from Nmap’s output
  \((-iX\ -oX)\) \((-iN\ -oN)\)
- Nmap notation in target/service specification
  e.g. 10.0.0-255.1-254, microsoft.com/24, 150.140.*.*
- High quality username/password lists
  (jtr, leaked phpbb/myspace etc)
- Platform portability: Windows, *BSD, Linux, Mac OS X
- --resume, --save
- IPv6 support, interactive output (Nmap style)
Resources

i. http://nmap.org/ncrack

$ svn co --username guest --password "" \
> svn://svn.insecure.org/ncrack