Security Analysis with Wireshark

By Russ McRee

It's not likely that you'll run into someone working in information security who hasn't heard of Ethereal.

As of May 2006, the world's most popular network protocol analyzer has become Wireshark. Gerald Combs, Ethereal's creator, was unable to reach agreement with his now former employer, which holds trademark rights to the Ethereal name. However, Combs wisely maintained ownership of the source code, and so Wireshark is, for all intents and purposes, Ethereal continued. Wireshark.org offers all the details of the name change and the Ethereal/Wireshark history, so we'll not cover any more of that here.

Wireshark/Ethereal is considered invaluable to network engineers for troubleshooting and analysis.

Understanding how to use this tool is certainly extremely useful in the infosec workplace, and essential for classes like SANS' Security 503: Intrusion Detection In-Depth, or most anything Laura Chappell's Protocol Analysis Institute offers. Wireshark/Ethereal is considered invaluable to network engineers for troubleshooting and analysis; and for security analysts there is deeply revealing insight at your fingertips. Keep in mind that, as we discuss Wireshark, the vast majority of our content is also immediately relevant to older versions of Ethereal.

As a network protocol analyzer, Wireshark offers the requisite GUI and TShark for text mode, display filters, live capture and offline analysis; reads a plethora of capture file formats; and supports hundreds of protocols. Wireshark works well on most any platform. The documentation is well written and will take you step by step through the process. I run it on Windows, two flavors of Linux, and Mac OS X. Just keep in mind the prerequisites for installation: on Linux/UNIX systems the GTK+ and libpcap packages must be installed, but the Wireshark installation package for Windows now includes WinPcap, so no extra effort required there. On Mac OS X, installation and function offer a bit of a challenge, a fact I can attest to thanks to the MacBook Pro I use. Wikipedia sums up the issue thus: "GTK+ only works with X11 on Mac OS X, so the user will need to run an X server"[1]. I've included some additional references for running Wireshark on Mac OS X at the end of this column.

As we proceed through our discussion we'll assume, given the widespread use and ease of installation, that you're running Wireshark on Windows.

Security issues

Before diving into Wireshark use scenarios, let's review some security issues innate to the tool itself.

First and foremost, the sheer nature of Wireshark makes it more vulnerable than other software, in part because it's written in ANSI C, and "the developers providing code to Wireshark ... have very divergent programming experience, from advanced networking specialists to novice programmers, making it more likely that new bugs get in"[2].

From the Wireshark wiki come four extremely valuable points:

- Always update to the latest Wireshark version available as bugs are fixed often.
- Don't run Wireshark as root/administrator. See http://wiki.wireshark.org/CaptureSetup/CapturePrivileges for details.
- Analyze capture files in an uncritical environment. Create a special (limited) user account or even use a dedicated machine for this task.
- Use a small capture tool which is less likely affected by security bugs, e.g.: tcpdump and transfer the capture file to the uncritical environment mentioned above[3].

There's no better way to persuade them than by showing how easy it is to pull their credentials off the wire.

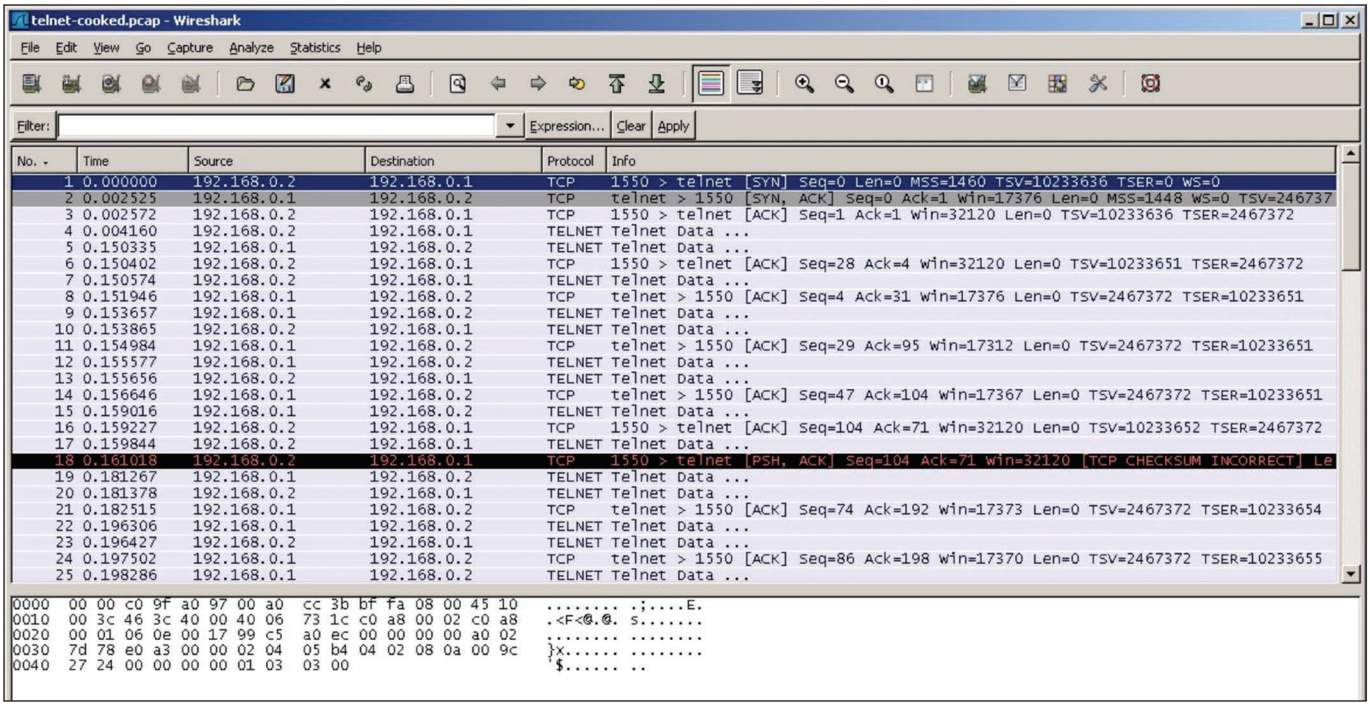
A simple scenario with Wireshark: Down with cleartext

We've all seen it: legacy servers and apps, older network devices, FTP, email apps. What information security evil do they all share? Cleartext protocols, one and all. Imagine the old-school DBA or legacy systems manager who has a hard time understanding what could possibly be wrong with telnet or R commands. Let's assume you have a SPAN port with a tap or a hub for network analysis, and have management buy-in to sniff traffic. You've been mandated with clearing out cleartext protocols, but need to drive the point home for hardcore legacy users upset by the premise of having to use an SSH client. There's no better way to persuade them than by showing how easy it is to pull their credentials off the wire.

[1] http://en.wikipedia.org/wiki/Wireshark

[2] http://wiki.wireshark.org/Security

[3] http://wiki.wireshark.org/Security


telnet-cooked.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

```
No. -  Time      Source        Destination   Protocol Info
     1 0.000000  192.168.0.2   192.168.0.1   TCP    1550 > telnet [SYN] Seq=0 Len=0 MSS=1460 TSV=10233636 TSER=0 WS=0
     2 0.002525  192.168.0.1   192.168.0.2   TCP    telnet > 1550 [SYN, ACK] Seq=0 Ack=1 Win=17376 Len=0 MSS=1448 WS=0 TSV=246737
     3 0.002572  192.168.0.2   192.168.0.1   TCP    1550 > telnet [ACK] Seq=1 Ack=1 Win=32120 Len=0 TSV=10233636 TSER=2467372
     4 0.004160  192.168.0.2   192.168.0.1   TELNET Telnet Data ...
     5 0.150335  192.168.0.1   192.168.0.2   TELNET Telnet Data ...
     6 0.150402  192.168.0.2   192.168.0.1   TCP    1550 > telnet [ACK] Seq=28 Ack=4 Win=32120 Len=0 TSV=10233651 TSER=2467372
     7 0.150574  192.168.0.2   192.168.0.1   TELNET Telnet Data ...
     8 0.151946  192.168.0.1   192.168.0.2   TCP    telnet > 1550 [ACK] Seq=4 Ack=31 Win=17376 Len=0 TSV=2467372 TSER=10233651
     9 0.153657  192.168.0.1   192.168.0.2   TELNET Telnet Data ...
    10 0.153865  192.168.0.2   192.168.0.1   TELNET Telnet Data ...
    11 0.154984  192.168.0.1   192.168.0.2   TCP    telnet > 1550 [ACK] Seq=29 Ack=95 Win=17312 Len=0 TSV=2467372 TSER=10233651
    12 0.155577  192.168.0.1   192.168.0.2   TELNET Telnet Data ...
    13 0.155656  192.168.0.2   192.168.0.1   TELNET Telnet Data ...
    14 0.156646  192.168.0.1   192.168.0.2   TCP    telnet > 1550 [ACK] Seq=47 Ack=104 Win=17367 Len=0 TSV=2467372 TSER=10233651
    15 0.159016  192.168.0.1   192.168.0.2   TELNET Telnet Data ...
    16 0.159227  192.168.0.2   192.168.0.1   TCP    1550 > telnet [ACK] Seq=104 Ack=71 Win=32120 Len=0 TSV=10233652 TSER=2467372
    17 0.159844  192.168.0.2   192.168.0.1   TELNET Telnet Data ...
    18 0.161018  192.168.0.2   192.168.0.1   TCP    1550 > telnet [PSH, ACK] Seq=104 Ack=71 Win=32120 [TCP CHECKSUM INCORRECT] Le
    19 0.181267  192.168.0.1   192.168.0.2   TELNET Telnet Data ...
    20 0.181378  192.168.0.2   192.168.0.1   TELNET Telnet Data ...
    21 0.182515  192.168.0.1   192.168.0.2   TCP    telnet > 1550 [ACK] Seq=74 Ack=192 Win=17373 Len=0 TSV=2467372 TSER=10233654
    22 0.196306  192.168.0.1   192.168.0.2   TELNET Telnet Data ...
    23 0.196427  192.168.0.2   192.168.0.1   TELNET Telnet Data ...
    24 0.197502  192.168.0.1   192.168.0.2   TCP    telnet > 1550 [ACK] Seq=86 Ack=198 Win=17370 Len=0 TSV=2467372 TSER=10233655
    25 0.198286  192.168.0.1   192.168.0.2   TELNET Telnet Data ...

0000  00 00 c0 9f a0 97 00 a0  cc 3b bf fa 08 00 45 10   ........ .;....E.
0010  00 3c 46 3c 40 00 40 06  73 1c c0 a8 00 02 c0 a8   .<F<@.@. s.......
0020  00 01 06 0e 00 17 99 c5  a0 ec 00 00 00 00 a0 02   ........ ........
0030  7d 78 e0 a3 00 00 02 04  05 b4 04 02 08 0a 00 9c   }x...... ........
0040  27 24 00 00 00 00 01 03  03 00                     '$...... ..
```

Figure 1. Default Wireshark view

Please, above all else, be certain you have approval to conduct this exercise. In the wrong circles, this activity can get you in a lot of trouble. When I've done this in the past, I've always informed the user in question of my intent and asked that they use a non-critical account or temporarily change their password, in the understanding that you're going to hand it back to them.

The recommended method is to capture the traffic with a smaller, safer tool like tcpdump or WinDump. Yes, we're focused on Wireshark, but you need to be proficient with basic captures too, so here's a crash course in tcpdump/WinDump. Get WinDump here: http://www.winpcap.org/windump/install/.

Assuming a Windows system is attached to the hub on your SPAN port, execute `windump -D` from a command prompt. This will tell you what interfaces are available. The interfaces will be sequentially numbered if there are more than one. For our example we'll use interface 2.

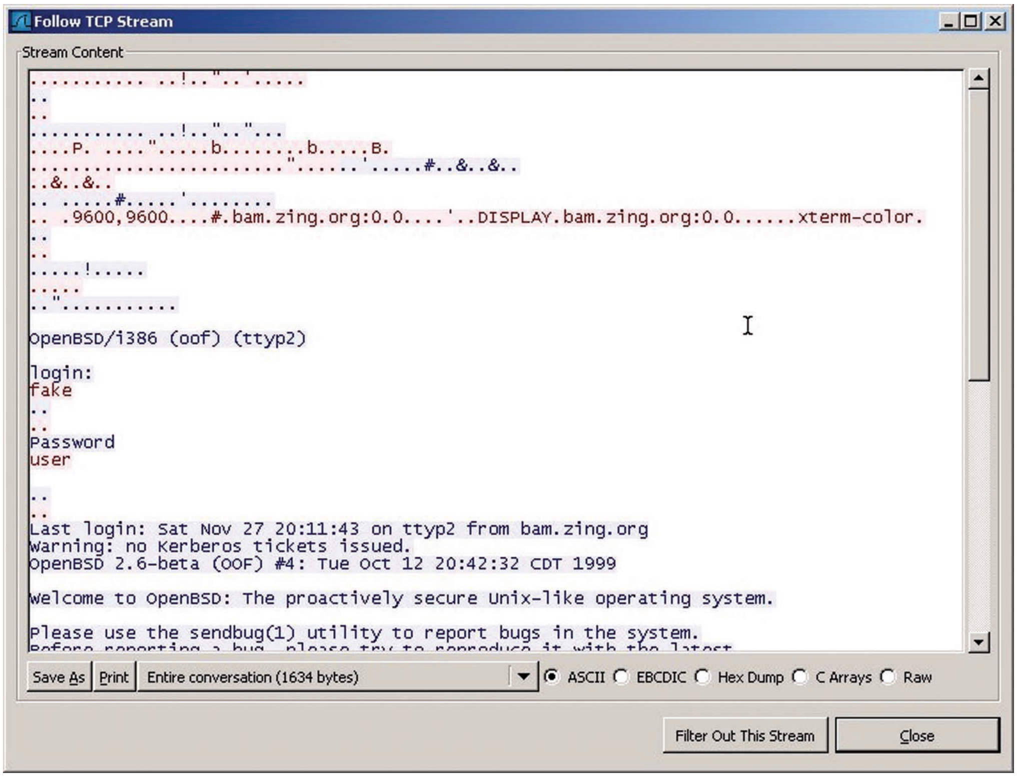
Issue the command `windump -i 2 > cleartext.pcap`.

This will write the capture to `cleartext.pcap` which you can then review in Wireshark. Have your user log on to their favorite legacy system via telnet while you are running the capture. Once they've completed a quick process, or just logged on or off, stop the WinDump capture and move the `cleartext.pcap` file to your machine where you're running Wireshark.

If you'd like to conduct the analysis part of this scenario as you read, pretend that a sample capture available on the Wireshark wiki is the file you've captured.

Grab the `telnet-cooked.pcap` at http://wiki.wireshark.org/SampleCaptures under the telnet section.

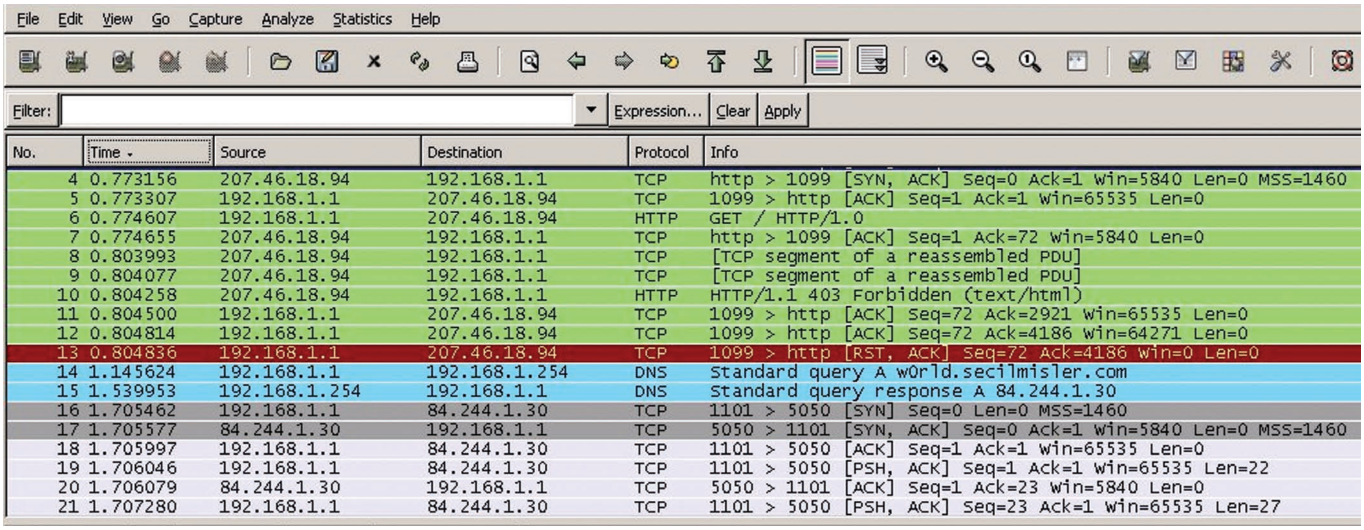
Open the capture file in Wireshark and it should look like Figure 1.



```
........... ..!.."..'.....
..
..
........... ..!.."..."...
....P. ....".....b........b.....B.
.........................."......'.....#..&..&..
..&..&..
.. .....#.....'........
.. .9600,9600....#.bam.zing.org:0.0....'..DISPLAY.bam.zing.org:0.0......xterm-color.
..
..
.....!.....
..;..
..".........

OpenBSD/i386 (oof) (ttyp2)

login:
fake
..
..
Password
user

..
..
Last login: Sat Nov 27 20:11:43 on ttyp2 from bam.zing.org
Warning: no Kerberos tickets issued.
OpenBSD 2.6-beta (OOF) #4: Tue Oct 12 20:42:32 CDT 1999

Welcome to OpenBSD: The proactively secure Unix-like operating system.

Please use the sendbug(1) utility to report bugs in the system.
Before reporting a bug, please try to reproduce it with the latest
```

Entire conversation (1634 bytes) ● ASCII ○ EBCDIC ○ Hex Dump ○ C Arrays ○ Raw

Figure 2. Follow TCP Stream reveals all



```
No.   Time ⌄      Source          Destination     Protocol  Info
    4 0.773156    207.46.18.94    192.168.1.1     TCP       http > 1099 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
    5 0.773307    192.168.1.1     207.46.18.94    TCP       1099 > http [ACK] Seq=1 Ack=1 Win=65535 Len=0
    6 0.774607    192.168.1.1     207.46.18.94    HTTP      GET / HTTP/1.0
    7 0.774655    207.46.18.94    192.168.1.1     TCP       http > 1099 [ACK] Seq=1 Ack=72 Win=5840 Len=0
    8 0.803993    207.46.18.94    192.168.1.1     TCP       [TCP segment of a reassembled PDU]
    9 0.804077    207.46.18.94    192.168.1.1     TCP       [TCP segment of a reassembled PDU]
   10 0.804258    207.46.18.94    192.168.1.1     HTTP      HTTP/1.1 403 Forbidden (text/html)
   11 0.804500    192.168.1.1     207.46.18.94    TCP       1099 > http [ACK] Seq=72 Ack=2921 Win=65535 Len=0
   12 0.804814    192.168.1.1     207.46.18.94    TCP       1099 > http [ACK] Seq=72 Ack=4186 Win=64271 Len=0
   13 0.804836    192.168.1.1     207.46.18.94    TCP       1099 > http [RST, ACK] Seq=72 Ack=4186 Win=0 Len=0
   14 1.145624    192.168.1.1     192.168.1.254   DNS       Standard query A w0rld.secilmisler.com
   15 1.539953    192.168.1.254   192.168.1.1     DNS       Standard query response A 84.244.1.30
   16 1.705462    192.168.1.1     84.244.1.30     TCP       1101 > 5050 [SYN] Seq=0 Len=0 MSS=1460
   17 1.705577    84.244.1.30     192.168.1.1     TCP       5050 > 1101 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
   18 1.705997    192.168.1.1     84.244.1.30     TCP       1101 > 5050 [ACK] Seq=1 Ack=1 Win=65535 Len=0
   19 1.706046    192.168.1.1     84.244.1.30     TCP       1101 > 5050 [PSH, ACK] Seq=1 Ack=1 Win=65535 Len=22
   20 1.706079    84.244.1.30     192.168.1.1     TCP       5050 > 1101 [ACK] Seq=1 Ack=23 Win=5840 Len=0
   21 1.707280    192.168.1.1     84.244.1.30     TCP       1101 > 5050 [PSH, ACK] Seq=23 Ack=1 Win=65535 Len=27
```

Figure 3. Malware capture

Here's where the fun begins. One of my favorite features in Wireshark (and Ethereal) is its ability to follow a TCP stream. In this case, if you highlight the very first packet and right-click it and select Follow TCP Stream (also available in the Analyze menu), another window will open displaying stream content from that packet's participation in a complete TCP conversation. Here's where you bring it home for your Mr. Telnet.

As you can see, Figure 2 clearly offers up the whole picture, including a username of "fake" and password of "user," nicely color-coded to distinguish client and server.

In fact, colorization is another key feature in Wireshark. If you select the View menu and choose Coloring Rules you'll quickly determine the default color of each packet type, and you can customize as you wish, including user-contributed color filters.

A more complex scenario with Wireshark: Analyzing malware behavior

One of my favorite tasks as a security wonk is infecting a Windows XP virtual instance with a variety of malware and watching what it does on the network. While the details of the environment I prefer to do this in might be of interest to you (CentOS, VMware, Snort, OllyDbg, etc.), and it's something I'll likely write about at a later date, we'll focus here on just Wireshark's role in the process.

NOTE: Do not do this anywhere near a production network, or any network of importance for that matter. Even if unleashing malware in a virtual environment, do it while established on a dedicated DSL connection designated for this purpose alone. You could disconnect from a network just prior to engaging the malware binary, but then you may miss some of its key behavior if it can't call home or query DNS. Please, be careful!

In a virtual environment, where you've infected your virtual instance, the host OS running the virtual server is the best place to run Wireshark for analysis of this nature. In this environment, because you've already taken the red pill and gone down the rabbit hole (thanks, Ed Skoudis), don't worry yourself with best practice – go ahead and capture directly with Wireshark rather than tcpdump/WinDump. Using VMware Server as an example, set the capture interface to VMnet8 to gather traffic from your infected guest OS. Go to Capture, then select your Interfaces, and then choose "capture" next to the VMnet8 interface to do so.

For this exercise, imagine you've been asked to analyze an executable called b0t.exe that a number of users have received in email.

On your unpatched, AV-less virtual guest execute b0t.exe while your Wireshark capture is running on the host OS.

It won't take much time, 60 seconds at most while connected to the Internet, to find out what b0t.exe might generate in the way of network traffic.

Then stop the capture and wait for it to load in the main view.

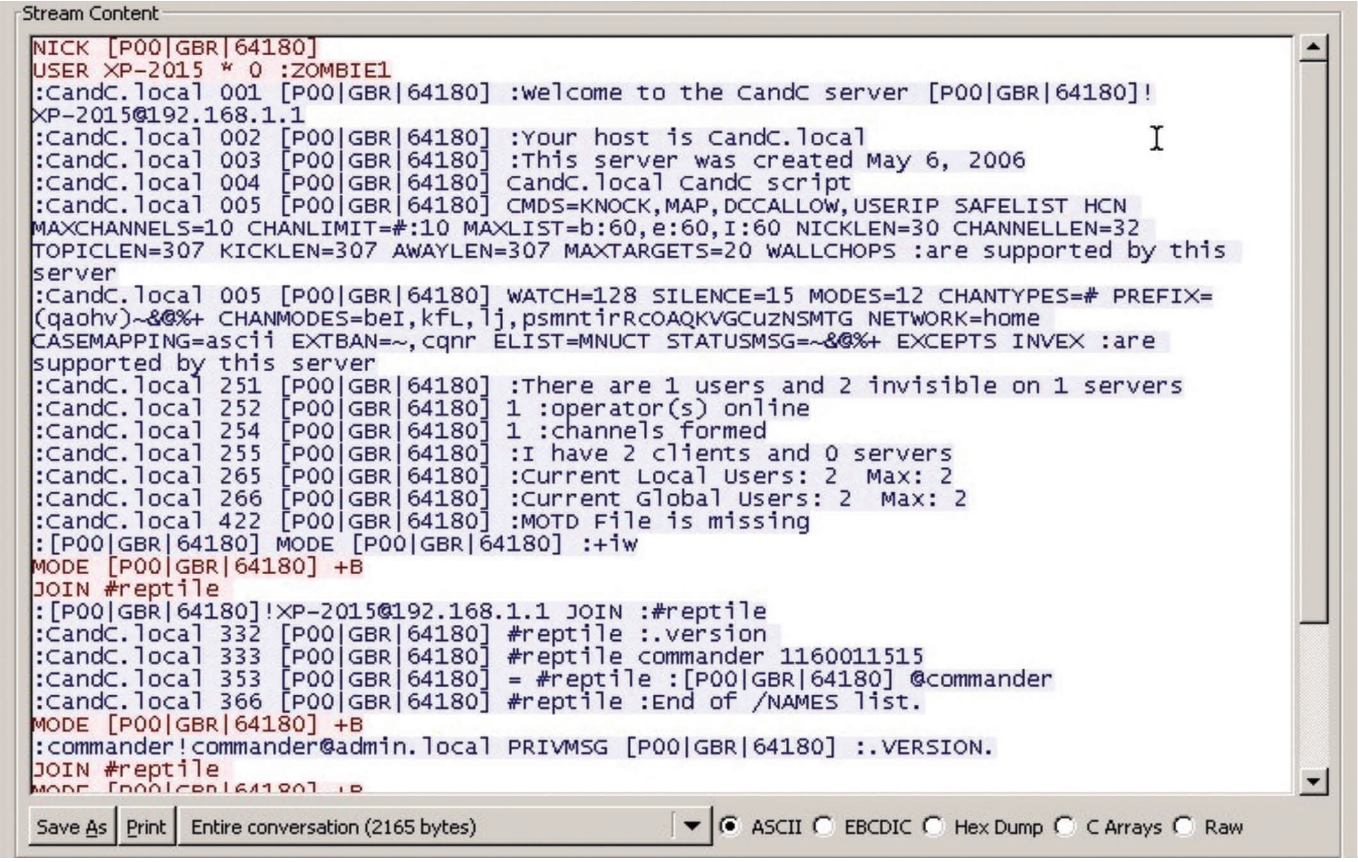
Again, you can follow along as you read by grabbing a copy of the very pcap file we're discussing from here: http://holisticinfosec.org/toolsmith/files/nov2k6/toolsmith.pcap.

No question, your virtual guest is now a bot and talking to a C&C (command and control) server.

In the first 16 packets, much is revealed. In #14 we see a DNS query to a strange domain name immediately followed by a response from 84.244.1.30. A quick WHOIS search reveals that this IP is in Amsterdam. The very next line really starts to clue you in. We see the local host making a call to 84.244.1.30 with a destination port of 5050. Google "port 5050" and we see a UDP reference (irrelevant, as your capture indicates TCP); the fact that Yahoo! Messenger might use TCP 5050 (interesting); but best of all, at the ISS site we learn that TCP 5050 is used by eggdrop, indicated as "the most popular bot." Now we're getting to the good stuff.

Let's revisit the Follow TCP Stream analysis method. Right-click on "packet 16" and choose Follow TCP Stream.

No question, your virtual guest is now a bot and talking to a C&C (command and control) server. But we still don't know exactly what bot we really have. Hit Clear on your Filter toolbar after you're done



```
NICK [P00|GBR|64180]
USER XP-2015 * 0 :ZOMBIE1
:CandC.local 001 [P00|GBR|64180] :Welcome to the CandC server [P00|GBR|64180]!
XP-2015@192.168.1.1
:CandC.local 002 [P00|GBR|64180] :Your host is CandC.local
:CandC.local 003 [P00|GBR|64180] :This server was created May 6, 2006
:CandC.local 004 [P00|GBR|64180] CandC.local CandC script
:CandC.local 005 [P00|GBR|64180] CMDS=KNOCK,MAP,DCCALLOW,USERIP SAFELIST HCN
MAXCHANNELS=10 CHANLIMIT=#:10 MAXLIST=b:60,e:60,I:60 NICKLEN=30 CHANNELLEN=32
TOPICLEN=307 KICKLEN=307 AWAYLEN=307 MAXTARGETS=20 WALLCHOPS :are supported by this
server
:CandC.local 005 [P00|GBR|64180] WATCH=128 SILENCE=15 MODES=12 CHANTYPES=# PREFIX=
(qaohv)~&@%+ CHANMODES=beI,kfL,lj,psmntirRcOAQKVGCuzNSMTG NETWORK=home
CASEMAPPING=ascii EXTBAN=~,cqnr ELIST=MNUCT STATUSMSG=~&@%+ EXCEPTS INVEX :are
supported by this server
:CandC.local 251 [P00|GBR|64180] :There are 1 users and 2 invisible on 1 servers
:CandC.local 252 [P00|GBR|64180] 1 :operator(s) online
:CandC.local 254 [P00|GBR|64180] 1 :channels formed
:CandC.local 255 [P00|GBR|64180] :I have 2 clients and 0 servers
:CandC.local 265 [P00|GBR|64180] :Current Local Users: 2  Max: 2
:CandC.local 266 [P00|GBR|64180] :Current Global Users: 2  Max: 2
:CandC.local 422 [P00|GBR|64180] :MOTD File is missing
:[P00|GBR|64180] MODE [P00|GBR|64180] :+iw
MODE [P00|GBR|64180] +B
JOIN #reptile
:[P00|GBR|64180]!XP-2015@192.168.1.1 JOIN :#reptile
:CandC.local 332 [P00|GBR|64180] #reptile :.version
:CandC.local 333 [P00|GBR|64180] #reptile commander 1160011515
:CandC.local 353 [P00|GBR|64180] = #reptile :[P00|GBR|64180] @commander
:CandC.local 366 [P00|GBR|64180] #reptile :End of /NAMES list.
MODE [P00|GBR|64180] +B
:commander!commander@admin.local PRIVMSG [P00|GBR|64180] :.VERSION.
JOIN #reptile
MODE [P00|GBR|64180] +B
```

Figure 4. Bot conversation revealed

with the TCP Stream analysis to return to the complete conversation.

We've learned a lot in just 16 packets, but what else might we find?

You'll start to see some HTTP GET requests as you scroll by packet 42, as well as more interesting DNS requests. Packet 95 really got it for me: a DNS request to www.kinchan.net – not good. Packet 111 gives it all away. Right-click that packet, Follow TCP Stream again, and you'll immediately see:

If you Google http://www.kinchan.net/cgi-bin/proxy.cgi you'll get a fairly immediate hit on W32/Tilebot-FV. Further research at the Sophos site quickly reveals that an alias for our little friend is W32/Sdbot.

Congratulations, you're the proud owner of an Sdbot variant, one of many polluting the Internet with ill intent.

Now let's do something about it to protect our users....

Firewall rules with Wireshark

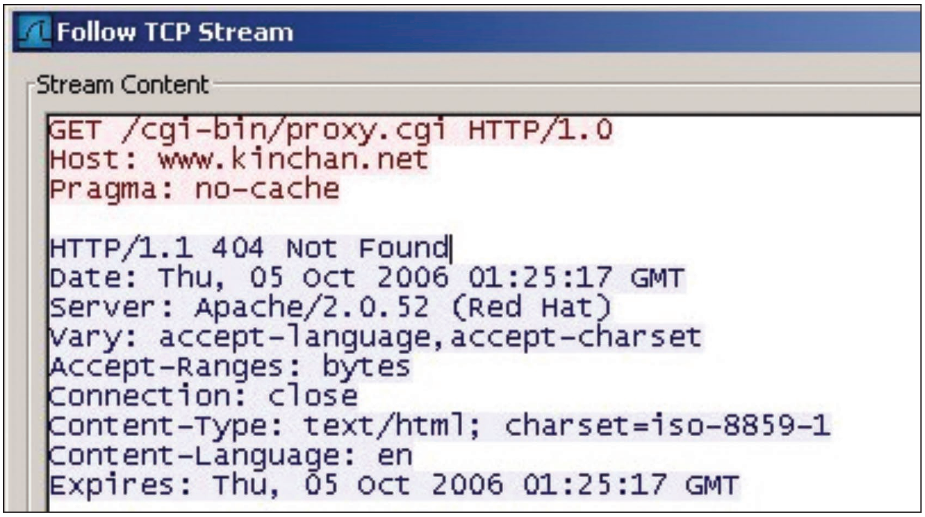
Amongst the plethora of functionality Wireshark includes is the ability to create firewall rules from a capture. Continuing on with our Sdbot pcap, highlight packet 17, choose Analyze and then Firewall ACL Rules.

You're in immediate luck if you're a Cisco shop. Choose Cisco IOS (extended) and you'll see:

There are a number of other options including IPFilter, ipfw, iptables, and even a Windows Firewall option. While it's typically not recommended to block lots of single IPs on your router (CPU utilization), you get the idea.

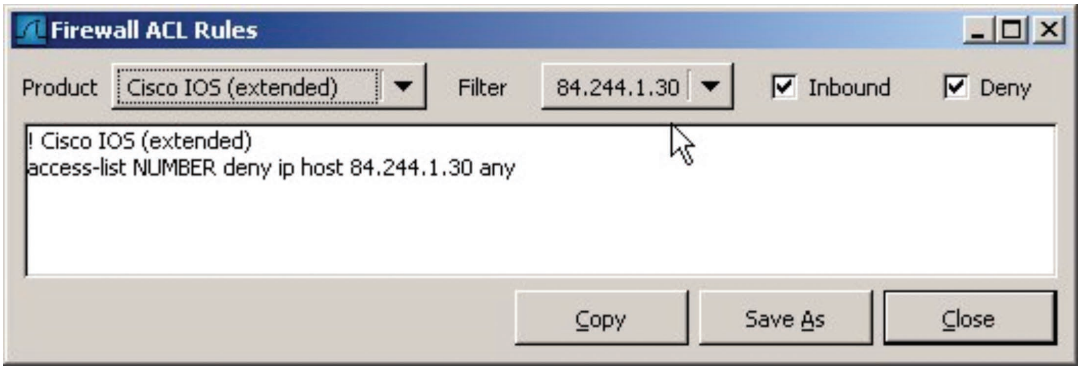
In closing

We have only touched on some very basic uses for Wireshark, particularly from an information security perspective, but nonetheless you can see the value of this application. It is, undoubtedly, one of the best open-



```
GET /cgi-bin/proxy.cgi HTTP/1.0
Host: www.kinchan.net
Pragma: no-cache

HTTP/1.1 404 Not Found
Date: Thu, 05 Oct 2006 01:25:17 GMT
Server: Apache/2.0.52 (Red Hat)
Vary: accept-language,accept-charset
Accept-Ranges: bytes
Connection: close
Content-Type: text/html; charset=iso-8859-1
Content-Language: en
Expires: Thu, 05 Oct 2006 01:25:17 GMT
```

Figure 5. Sdbot variant



```
! Cisco IOS (extended)
access-list NUMBER deny ip host 84.244.1.30 any
```

Figure 6. Wireshark Firewall ACL Rules

source tools you could ever ask for in your endeavor to protect your user base and understand your network.

As always when using tools like Wireshark, remember – have permission, be diligent, and be true to our cause.

See you next month.

About the Author

Russ McRee, GCIH, is a security analyst working in the Seattle area. He is a member of ISSA, Pacciso, InfraGard, and CCSA (Cyber Conflict Studies Association). Russ maintains holisticinfosec.org. Contact him at russ@holisticinfosec.org.

References

Working out the X11 issue for Wireshark on Mac OS X:

http://www.mail-archive.com/wireshark-users@wireshark.org/msg00356.html

After you've worked out X11 and you need to establish interfaces: http://www.ethereal.com/lists/ethereal-users/200608/msg00021.html