

Security Alert

Capturing Peer-to-Peer Applications

In the September issue of *Novell Connection*, I blew the whistle on Gnutella, a peer-to-peer file-sharing protocol that has gained a worldwide audience. (See “Security Alert: Just Say Gno!” pp. 33–35. You can download this article from www.ncmag.com/past.) Using Gnutella, users can share files (such as MP3, software, image, audio, and video files) across the Internet. Gnutella is available at www.limewire.com, www.bearshare.com, and www.eBlvd.com.

As I explained in the September issue, Gnutella is a risk your company cannot afford, and you should completely eradicate Gnutella from your company’s network. Specifically, Gnutella creates the following problems:

- **Bandwidth Overhead.** Gnutella hosts, which are called *servents*, establish a TCP connection with each of the other servents on the Gnutella network. After the connection is made, the other Gnutella servents send their list of searches throughout the Gnutella network. This traffic can run between 4,500 and 5,300 bytes per second.
- **Security Risks.** A Gnutella servent can share any local files the user makes available—including any network files to which the user has rights. In addition, the Gnutella protocol specification can bypass firewalls that are set up to block file downloads. For example, users inside a company can configure the Gnutella application to use ports that are allowed through the firewall, such as port 80 (HTTP), 443 (HTTPS), 23 (Telnet), 25 (SMTP), or 110 (POP3).
- **Virus Vulnerabilities.** Gnutella is fertile ground for computer viruses. In February 2001, the Mandragore worm infected the Gnutella network and was spread to connected computers.

In the September issue, I also explained how you can build a filter to capture Gnutella connection sequences and file transfer sequences. Since I wrote that article, I’ve had time to work with two other popular peer-to-peer applications: Morpheus and iMesh. Like Gnutella, Morpheus and iMesh create unnecessary traffic on your company’s network and compromise network security. This article explains how to build filter patterns that will catch these lousy applications.



WHAT ARE FILTERS?

A filter is a set of criteria that a packet must match to be accepted in the trace buffer or to be displayed. You, as the protocol analyst, must create a set of filters that match the traffic you are interested in viewing. Don’t count on your protocol analyzer having a complete set of prebuilt filters. The vendors have been noticeably lax in supplying strong filters out of the box.

You need to know two things to build really great filters:

- The offset that indicates where you are looking in the packet
- The value that you are looking for at that offset

This article provides the offsets and the values to capture both Morpheus and iMesh traffic. (These filters assume that you are using the Ethernet II frame type over an Ethernet network.) In addition, the filters in this article are created on Sniffer Pro from Network Associates. If you are using another analyzer, you may need to change the offsets to decimal format.

CATCHING MORPHEUS

Morpheus is available at the following locations: www.musiccity.com and www.kazaa.com. Although not as bandwidth-intensive as Gnutella, Morpheus poses a security risk nonetheless. You can catch Morpheus traffic using two standard methods: You can either look for traffic to and from the Morpheus default port 1214, or you can filter on two known traffic signatures:

- Domain Naming System (DNS) query for supernode3.streamcastnetworks.com
- DNS query for supernode.kazaa.com

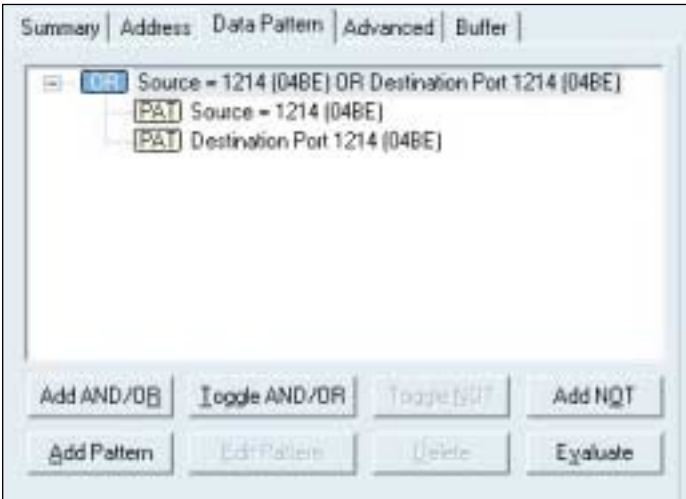


Figure 1. Because Morpheus does not port-roam yet, you can start capturing Morpheus traffic with a simple port number filter.



Figure 2. This filter is for Morpheus traffic. The hexadecimal value 0x04BE is equivalent to the decimal value 1214.

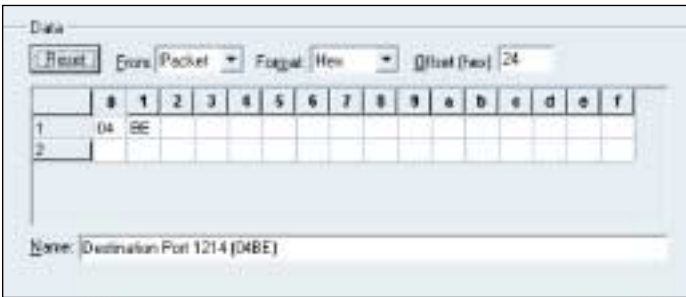


Figure 3. The destination port offset for Morpheus traffic is 0x24. (The source port offset is 0x22.)

Figures 1 through 3 show the filters used to catch Morpheus traffic based on the default Morpheus port number 1214 (decimal) or 0x04BE (hexadecimal). Figures 4 through 6 show the filters used to catch the DNS queries.

Now let's try creating more advanced filters—filters based on the application-layer packets that are unique to Morpheus. The Morpheus client makes two DNS queries at startup. By building a filter for either of these DNS queries, you can detect when a Morpheus application has started up on your company's network, regardless of the port number the application is using.

Note. At some point, Morpheus will probably start port-roaming. Morpheus may also change the DNS queries that it sends. You should be on the lookout so that you can detect



Figure 4. To catch Morpheus traffic based on the application-level calls, you build a filter that looks for the Morpheus client's DNS queries.

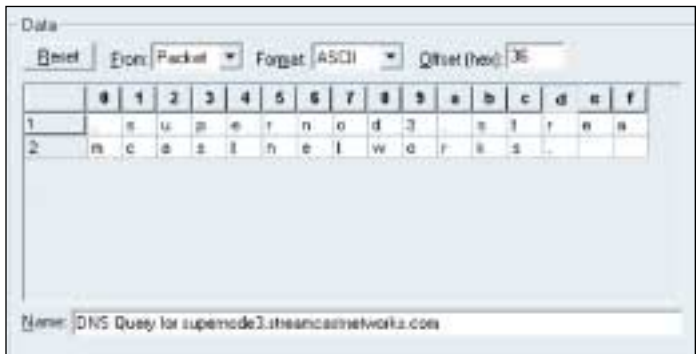


Figure 5. With Morpheus, each DNS query starts with a leading "." and runs two lines long. This query was abbreviated.

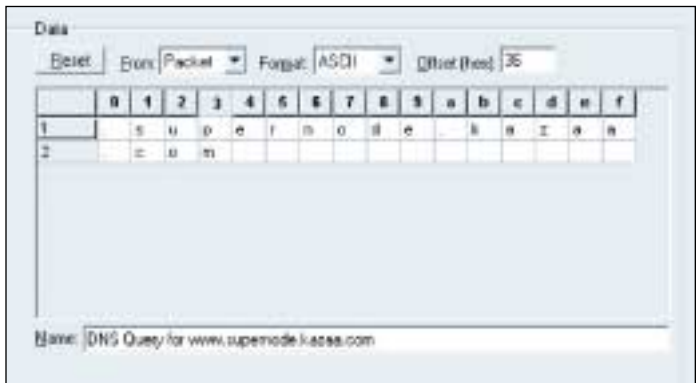


Figure 6. With Morpheus, the second DNS query is shorter; you can see that it identifies "kazaa."

these changes and continue to block Morpheus traffic on your company's network.

Figure 4 shows the summary view of the DNS queries filter. Remember to use the OR operand—not the AND operand. The OR operand means that the incoming packet must match either the first pattern or the second pattern. You would use the AND operand if you were looking for a packet that matches both filters.

Visit our advertiser, Novell, at www.novell.com

Visit our advertiser, Novell, at www.novell.com

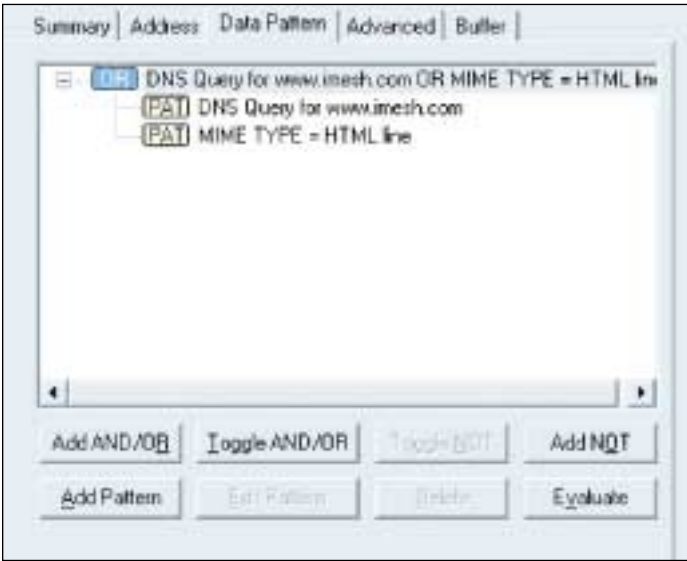


Figure 7. To catch iMesh traffic, you build a filter that looks for the iMesh DNS query or one of the first commands.



Figure 8. With iMesh, each DNS query starts with a leading “. ” (Unlike Morpheus, the query does not run two lines.)

Figures 5 and 6 show the DNS query patterns that we’re interested in. Notice that all of the DNS queries start with a leading period. You can filter on any DNS queries based on these offsets with different names inside. That’s not too difficult, eh?

Note. You may consider setting up a firewall to block access to each of the web sites listed in the DNS queries. However, you won’t catch the Morpheus clients if the DNS sites change.

CATCHING IMESH

Interestingly, iMesh (which is available at www.imesh.com) does not seem to use a specific set of port numbers. Instead, iMesh starts port-roaming from the moment it tries to connect to the iMesh server. As a result, iMesh poses a greater security threat than either Gnutella or Morpheus. Like Morpheus, however, iMesh is less bandwidth-intensive.

To capture iMesh, you must look for its two application-level signatures. Figures 7 through 9 show the filters used to catch iMesh traffic on the network.

The first filter is designed to catch the iMesh DNS query for www.imesh.com. Although you may consider building a web site filter to stop your company’s users from visiting



Figure 9. The only command that can be identified in the iMesh traffic is a reference to the “mime type.”

iMesh.com, such a filter won’t work. The DNS response indicates that the web site to which users connect can change; users don’t actually connect to www.imesh.com.

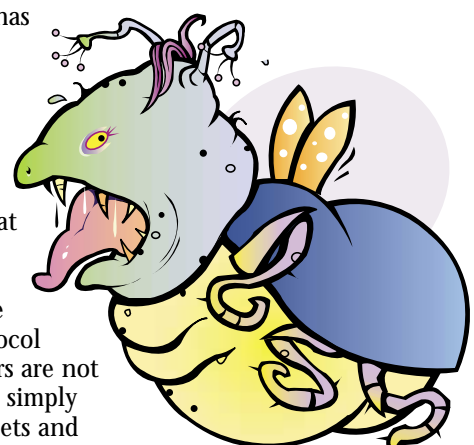
Once again, you use an OR operand in this filter. You want to catch packets that match the DNS query or the application command that starts with “MIME TYPE,” as shown in Figure 7.

Figures 8 and 9 show the specific patterns you are looking for. The first is the DNS query for www.imesh.com. The second filter looks for the pattern “MIME TYPE = HTML.”

If you are using Sniffer Pro, you must press the space bar to insert the spaces between words and the equal sign. If you skip the blank spaces, Sniffer Pro will clip off the remainder of the filter line.

CONCLUSION

Every application has some type of signature—something that distinguishes that application from other traffic on the network. Building a strong set of filters that takes advantage of application signatures enables you to get the most out of your protocol analyzer. Pattern filters are not difficult to make: You simply need to know the offsets and values that match your target packet.



Using these two elements, you can catch any type of packet that crosses the network.

Laura Chappell performs onsite network analysis sessions for troubleshooting, optimization, and security checks. She also teaches hands-on courses on protocol analysis. For more information about these services or courses, send an e-mail message to lchappell@packet-level.com.

To learn more about packet filtering using Network Associates’ Sniffer Pro or Wild Packets Inc. EtherPeek, you can read Packet Filtering: Catching the Cool Packets, Chappell’s new book which is available at www.podbooks.com.