Packet Capture Functionality

HOW TO GUIDE

February, 2011

Revision 1.4

Table of Contents

Figures

Excerpts

1. Introduction:

The WiNG platform includes many troubleshooting and diagnostic features to help the administrator in determining root cause and effect of many various problems that inhibit communications. The robust troubleshooting feature set built in to Zebra's WiNG software affords network administrators the ability to troubleshoot at any point in the WLAN infrastructure and the edge network.
One of these features is the built-in packet capture function, which is a command-line based tool. The purpose of this paper is to introduce the reader to the "pktcap" and "remote-debug" commands that allow traces to be captured and analyzed. It also gives a brief overview of some of the functionality this powerful command gives the network administrator.

1.1 Overview:

In WiNG 5.x, the "pktcap" command becomes much more robust in functionality. It now includes the ability to:

Capture at the core or the edge of the wireless network (i.e. – RFS switch and / or the access points)

Capture on any physical interface (Ethernet, radio, radio interface module (rim), etc.)

Capture on any logical interface (bridge, vlan, wireless, vpn, etc.)

Capture based on applied rules (deny, drop)

Capture to to remote destinations (tftp, ftp or a Tazman Sniffer Protocol (tzsp) host via the remote-debug command)

WiNG 5 now gives a network administrator fully distributed packet capture capabilities to perform troubleshooting at a very granular level. The following diagram represents points at which packet capture can be executed as related to an access point running WiNG 5; every logical and physical boundary can facilitate captures in both inbound and outbound directions:

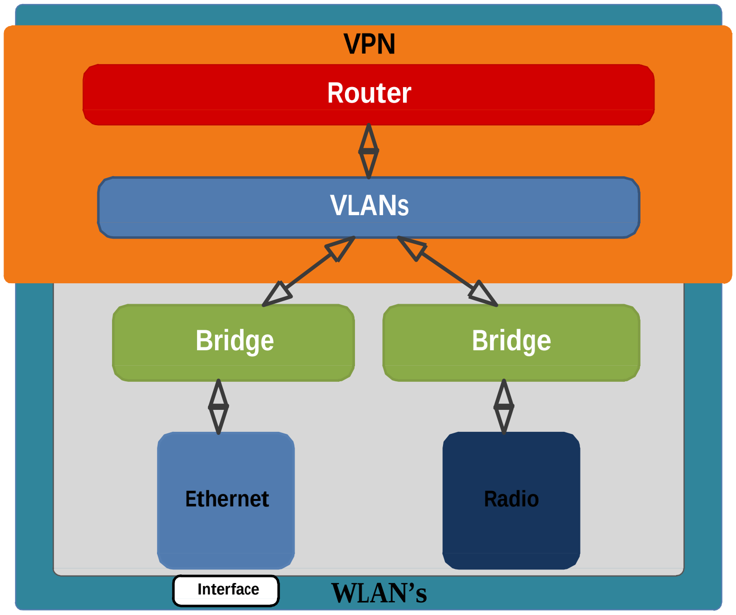


Figure 1: Points of Capture Logical Diagram

2. Pre-Requisites:

2.1 Requirements:

As of this writing, the following requirements must be met in order to utilize the packet capture feature described herein:

An RFS switch running WiNG version 5.0.0.0-107R or later.

AP650's or AP71731's running version 5.0.0.0-107R or later.

2.2 Components Used:

The information in this document is based on the following Zebra hardware and software versions:

1 x RFS4010 running version 5.0.0.0-107R.

1x AP7131 running version 5.0.0.0-107R.

1x Laptop

1x mobile client – in this case, an Apple iPhone 4

✓ Registered users may download the latest software and firmware from the Zebra Technical Support Site http://support.symbol.com.

3. Output Options:

By default, the "pktcap" feature will capture to the terminal / console from which it was started in real time. It will capture up to 50 packets, scrolling them by on the console. However this behavior is configurable; options include:

Save to local file in flash; file can be named

Locally saved captures can be up to 1M packets

Save to a tftp or ftp host

Capture to a remote TZSP host for real-time analysis with applications such as Wireshark. This is further discussed in section 6 (Advanced Capabilities).

3.1 CLI Configuration Example:

1)

```
RFS4000# service pktcap on interface vlan10 write pktcap.pcap
Capturing up to 50 packets. Use Ctrl-C to abort.
..
```

2)

```
RFS4000# dir
      Directory of flash:/.
```

```
drwx                 Mon Sep  6 09:49:29 2010   log
drwx                 Fri Dec 31 17:00:08 1999   configs
drwx                 Fri Dec 31 17:00:08 1999   cache
drwx                 Fri Dec 31 17:00:08 1999   crashinfo
-rw-    5476         Mon Sep  6 11:08:30 2010   pktcap.pcap

drwx                 Fri Dec 31 17:00:08 1999   hotspot
```

4. Configuration:

The following diagram depicts the setup:

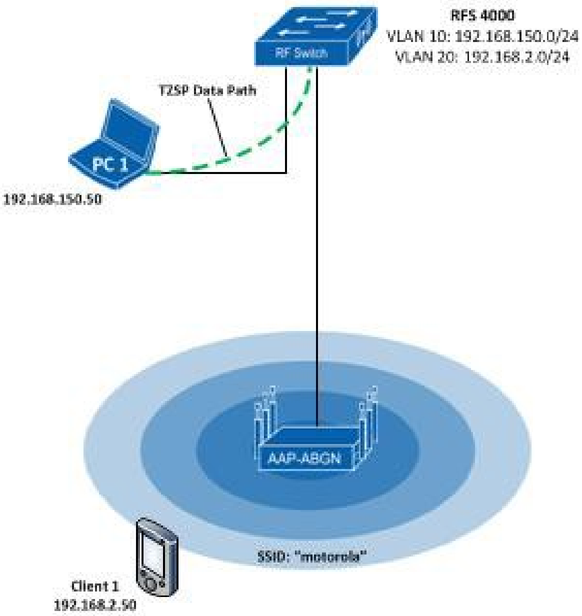


Figure 2: Scenario Diagram

4.1 Command Syntax:

“Pktcap” is a sub-command to the “service” command, as seen below:

```
      rfs4000-22A3AC#service ?
        advanced-wips       Advanced WIPS service commands
        clear               Reset functions
        cli-tables-expand   Expand the cli-table in drapdown format
        ..
        .
        mint                MiNT protocol
        pktcap              Start packet capture
        pm                  Process Monitor
        radio               Radio parameters
        ..
        .
```

```
rfs4000-22A3AC#service pktcap on ?
  bridge     Capture at bridge
  deny       Capture at deny locations
  drop       Capture at drop locations
  interface  Capture at an interface
  radio      Capture at radio (802.11)
  rim        Capture at radio interface module
  router     Capture at router
  vpn        Capture at vpn
  wireless   Capture at wireless to wired interface
```

As can be seen via the help feature, there are many options right away to use with the “pktcap” command. The packet capture feature of WiNG 5 allows the administrator to save the capture by various methods so that it can be reviewed later using a capture / decoding application such as Wireshark.

4.1.1 What and Where

One of the first questions that may come up for many administrators is *where* do I capture to see the data I want or *what* data will I see at “*named*” interface? These are good questions and to answer, it helps to logically think about the flow of data from the wireless client to the point that it enters the supporting wired infrastructure. A couple examples are included below:

4.1.1.1 At the Access Point

From client to AP, layer-2 connectivity data can be captured. Packets between immediate L2 devices (wireless client mac, data switch mac where AP is plugged in) and the mac-address of the BSS (particular SSID) are seen, and these packets are encrypted (if using encryption on the WLAN).

Excerpt 1: Radio Interface Capture

1) AP “Radio” Interface capture

```
AP7131# service pktcap on radio (1|2)

Capturing up to 50 packets. Use Ctrl-C to abort.
1 2:27:46.626429 O  ENCRYPT DATA  Src:00-23-68-22-A3-AC Dst:01-A0-F8-00-00-00 Bss:00-23-68-93-48-
61
2 2:27:47.241320 O  ENCRYPT DATA  Src:68-B5-99-E8-B2-7A Dst:FF-FF-FF-FF-FF-FF Bss:00-23-68-93-48-
61
3 2:27:47.241558 I  ENCRYPT QOS_DATA  Src:00-24-D7-60-DF-68 Dst:68-B5-99-E8-B2-7A Bss:00-23-68-
93-48-61
4 2:27:47.242400 O  ENCRYPT QOS_DATA  Src:68-B5-99-E8-B2-7A Dst:00-24-D7-60-DF-68 Bss:00-23-68-
93-48-61
```

To capture packets as they leave the AP, headed either for the infrastructure or the controller, capture at the “wireless” interface, which is from the wireless medium to the wired medium. At this point there will be wireless infrastructure overhead packets (MiNT) as well as the unencrypted data between wireless clients and the infrastructure.

Excerpt 2: Wireless Interface Capture

1) AP “Wireless” Interface capture

```
AP7131# service pktcap on wireless

Capturing up to 50 packets. Use Ctrl-C to abort.
1 2:15:35.612844 I ICMP: 192.168.150.11 > 192.168.150.100 echo request, id 512, seq 57465, length
40
2 2:15:35.613078 O ICMP: 192.168.150.100 > 192.168.150.11 echo reply, id 512, seq 57465, length
40
```

```
3 2:15:35.869095 O ARP: Who has 192.168.150.50? tell 192.168.150.7
4 2:15:36.613360 I ICMP: 192.168.150.11 > 192.168.150.100 echo request, id 512, seq 57721, length
40
5 2:15:36.613592 O ICMP: 192.168.150.100 > 192.168.150.11 echo reply, id 512, seq 57721, length
40
6 2:15:37.613775 I ICMP: 192.168.150.11 > 192.168.150.100 echo request, id 512, seq 57977, length
40
```

4.1.1.2 At the RFS (Controller)

Unless the controller includes built-in radios (i.e. rfs4011), then capturing at the “radio” or “wireless” interfaces will yield nothing; the commands will be accepted, but you’ll e waiting quite a while to see any packets as none are being capture.

On the RFS, it makes sense to capture on physical interfaces (ge*x*, up1), L3 logical interfaces (interface vlan1) or the *bridge* interface. And depending on which point you choose, you will see MiNT infrastructure packets and / or unencrypted client data packets. In the first example below, the AP7131 is plugged directly into an rfs4010. Thus, we can expect to see client data packets, broadcast / multicast packets from the AP as well as the controller and also direct communication MiNT packets between the AP and the controller (not all types were included in the excerpt).

Excerpt 3: RFS Ge1 Interface Capture

1) RFS “Ge1” Interface Capture

```
RFS4000# service pktcap on interface ge1

Capturing up to 50 packets. Use Ctrl-C to abort.
1 2:44:57.639687 I ICMP: 192.168.150.11 > 192.168.150.7 echo request, id 512, seq 46976, length
40
2 2:44:57.640284 O ICMP: 192.168.150.7 > 192.168.150.11 echo reply, id 512, seq 46976, length 40
3 2:44:57.963710 O MINT  00-23-68-22-A3-AC > 01-A0-F8-00-00-00 | DGRAM 68.22.A3.AC/0 >
00.00.00.00/12 router
4 2:44:58.263696 O MINT  00-23-68-22-A3-AC > 01-A0-F8-00-00-00 | DGRAM 68.22.A3.AC/0 >
00.00.00.00/12 router
5 2:44:58.639721 I ICMP: 192.168.150.11 > 192.168.150.7 echo request, id 512, seq 47232, length
40
6 2:44:58.640280 O ICMP: 192.168.150.7 > 192.168.150.11 echo reply, id 512, seq 47232, length 40
7 2:44:59.639750 I ICMP: 192.168.150.11 > 192.168.150.7 echo request, id 512, seq 47488, length
40
8 2:44:59.640403 O ICMP: 192.168.150.7 > 192.168.150.11 echo reply, id 512, seq 47488, length 40
9 2:45:00.626974 I MINT  00-23-68-93-13-CC > 01-A0-F8-00-00-00 | DGRAM 68.93.13.CC/0 >
00.00.00.00/12 router
```

Note: Notice that direction of the packet is indicated by (I | O) after the timestamp and before the protocol type.

Of course, on a L3 interface, we can expect to see L3 data. The following example shows data packets from the HTTP GUI on the RFS, various ARP data as well as the ICMP echo / reply packets from a client to the controller

Excerpt 4: L3 Interface Capture

1) RFS “vlan1” L3 Interface Capture

```
RFS4000# service pktcap on interface vlan1

9 3:00:32.561239 I ICMP: 192.168.150.11 > 192.168.150.100 echo request, id 512, seq 23684, length
40
10 3:00:32.561384 O ICMP: 192.168.150.100 > 192.168.150.11 echo reply, id 512, seq 23684, length
40
11 3:00:32.884424 O TCP: 192.168.150.100 > 192.168.150.7 ports 80 > 4431, data length 317, PA, DF
12 3:00:32.988723 I ARP: Who has 192.168.150.100? tell 192.168.150.7
13 3:00:32.988736 O ARP: 192.168.150.100 is at 00-23-68-22-A3-AC
```

```
14 3:00:32.989178 I TCP: 192.168.150.7 > 192.168.150.100 ports 4431 > 80, A, DF
15 3:00:33.020584 I TCP: 192.168.150.7 > 192.168.150.100 ports 4431 > 80, data length 798, PA, DF
16 3:00:33.020720 O TCP: 192.168.150.100 > 192.168.150.7 ports 80 > 4431, A, DF
```

4.1.1.3 Remote-Dubug “Distributed” Capture

Discuss more later, the remote-debug feature allows for promiscuous capture of all traffic a device hears. It is similar to having a “distributed” capture system in place. In this way, if one wishes to see what kind of traffic is in the air in a specific area, then this method can be used to capture all the packet in the air as heard by the device that remote-debug is executed on.

The difference is that “pktcap” captures traffic traversing a specific device and so the clients associated to that device will be seen as well as traffic sourced from or destined to the particular device that “pktcap” is executed at. With remote-debug live-cap executed at an access-point, traffic from any client heard by that device will be captured, whether the client is associated to that access point or not. Additionally, the access point may pick up management frames (probe requests, responses, beacons, etc.) from neighboring WLAN’s in other buildings, for other companies.

4.1.2 CLI Configuration Example:

Connecting to the AP7131 in the diagram and issuing a simple wireless capture

Excerpt 5: Wireless Interface Capture

1) Connecting to AP / Wireless Interface Capture

```
RFS4000# connect ap7131-85CD20
<output removed for brevity>
Ap7131-85CD20# service pktcap on wireless
Capturing up to 50 packets. Use Ctrl-C to abort.
1 19:37:32.023070 I WNMP-Config  DC-2B-61-16-CF-86 > 01-A0-F8-F0-F0-04
2 19:37:32.080950 I ARP: Who has 192.168.150.1? tell 192.168.150.48
3 19:37:32.081053 O ARP: 192.168.150.1 is at 00-23-68-22-D2-6E
4 19:37:32.081157 I ARP: Who has 10.0.0.1? tell 10.0.0.19
5 19:37:32.081166 O ARP: Who has 10.0.0.1? tell 10.0.0.19
6 19:37:32.082874 I UDP: 0.0.0.0 > 255.255.255.255 ports 68 > 67, data length 308
7 19:37:32.082928 O UDP: 0.0.0.0 > 255.255.255.255 ports 68 > 67, data length 308
8 19:37:32.083637 O UDP: 192.168.150.1 > 255.255.255.255 ports 67 > 68, data length 308, tos 0x10
```

In the above example, a packet capture to the console was started and then the iPhone client was connected to the WLAN (only the first 8 packets were included in the text).

In the following example, a simple capture to a file of the same action was performed, but this time at the RFS4000 switch:

Excerpt 6: Write to File Example

1) Write to File Example

```
rfs4000-22D26E#service pktcap on inter ge1 write example-cap.pcap count 20
filter ether host DC-2B-61-16-CF-86  - note the use of the filter
Capturing up to 20 packets. Use Ctrl-C to abort.
20
rfs4000-22D26E#dir
Directory of flash:/.

  drwx             Thu Sep 30 19:30:55 2010   log
  drwx             Sat Jan  1 00:00:08 2000   configs
  drwx             Sat Jan  1 00:00:08 2000   cache
  -rw-   5425      Wed Sep 29 21:06:20 2010   vlab-config
  drwx             Sat Jan  1 00:00:08 2000   crashinfo
  -rw-   3375      Thu Sep 30 19:45:06 2010   example-cap.pcap
  drwx             Sat Jan  1 00:00:08 2000   hotspot
  -rw-   24        Fri Sep 17 19:36:58 2010   wifi-pktcap-112310.pcap
```

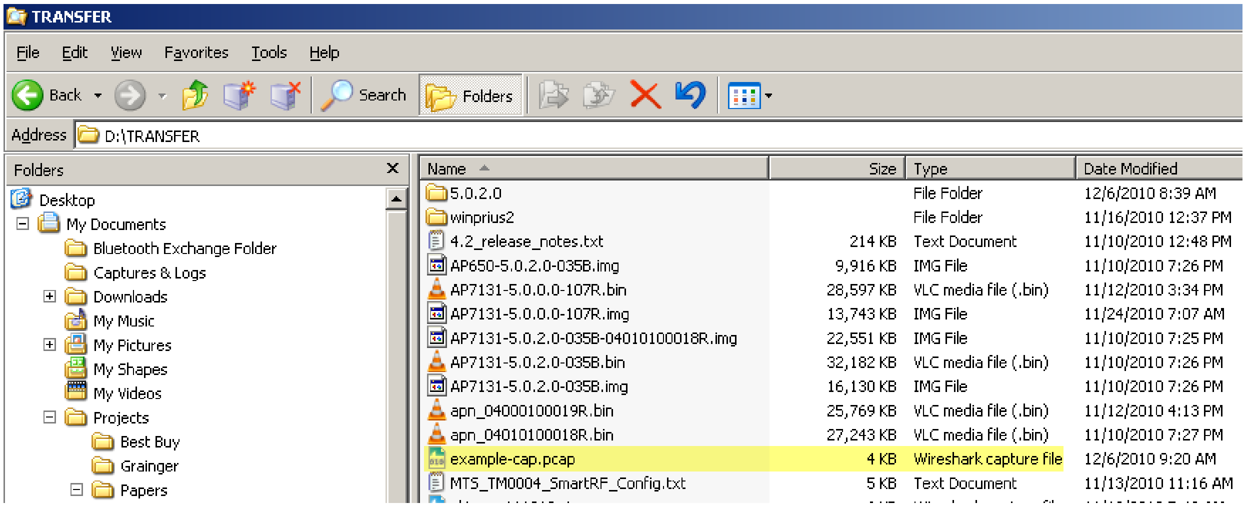
And to view the file using Wireshark, we simple copy the file to our laptop using tftp (tftpd32.exe (http://tftpd32.jounin.net/) running in this example):

Excerpt 7: Send to TFTP

1)	Sending Capture File to TFTP Server

```
rfs4000-22D26E#copy example-cap.pcap tftp://192.168.150.5/
```

We can then open the file to view the contents:



Address: D:\TRANSFER

- Desktop
  - My Documents
    - Bluetooth Exchange Folder
    - Captures & Logs
    - Downloads
    - My Music
    - My Pictures
    - My Shapes
    - My Videos
    - Projects
      - Best Buy
      - Grainger
      - Papers

Name	Size	Type	Date Modified
5.0.2.0		File Folder	12/6/2010 8:39 AM
winprius2		File Folder	11/16/2010 12:37 PM
4.2_release_notes.txt	214 KB	Text Document	11/10/2010 12:48 PM
AP650-5.0.2.0-035B.img	9,916 KB	IMG File	11/10/2010 7:26 PM
AP7131-5.0.0.0-107R.bin	28,597 KB	VLC media file (.bin)	11/12/2010 3:34 PM
AP7131-5.0.0.0-107R.img	13,743 KB	IMG File	11/24/2010 7:07 AM
AP7131-5.0.2.0-035B-04010100018R.img	22,551 KB	IMG File	11/10/2010 7:25 PM
AP7131-5.0.2.0-035B.bin	32,182 KB	VLC media file (.bin)	11/10/2010 7:26 PM
AP7131-5.0.2.0-035B.img	16,130 KB	IMG File	11/10/2010 7:26 PM
apn_04000100019R.bin	25,769 KB	VLC media file (.bin)	11/12/2010 4:13 PM
apn_04010100018R.bin	27,243 KB	VLC media file (.bin)	11/10/2010 7:27 PM
example-cap.pcap	4 KB	Wireshark capture file	12/6/2010 9:20 AM
MTS_TM0004_SmartRF_Config.txt	5 KB	Text Document	11/13/2010 11:16 AM

Figure 3: Example .pcap File

5. Useful Options:

5.1 Example 1 – Filters:

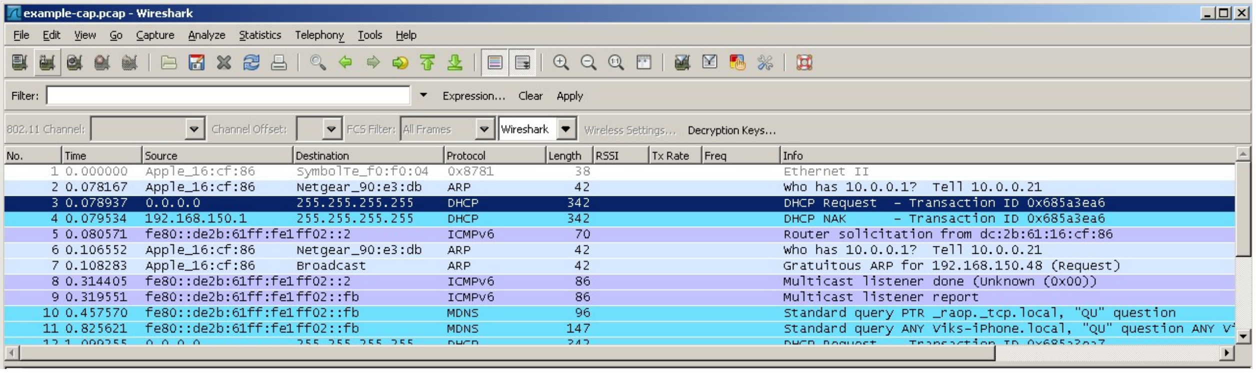
Continuing with our example, there are a number of very useful options an administrator can use while performing troubleshooting with the packet capture functions. Some of these include on-the-fly filters and the ability to limit how much of a packet is captured. The list of filter options follows:

Excerpt 8: Filter Options

1)	Pktcap Filter Options

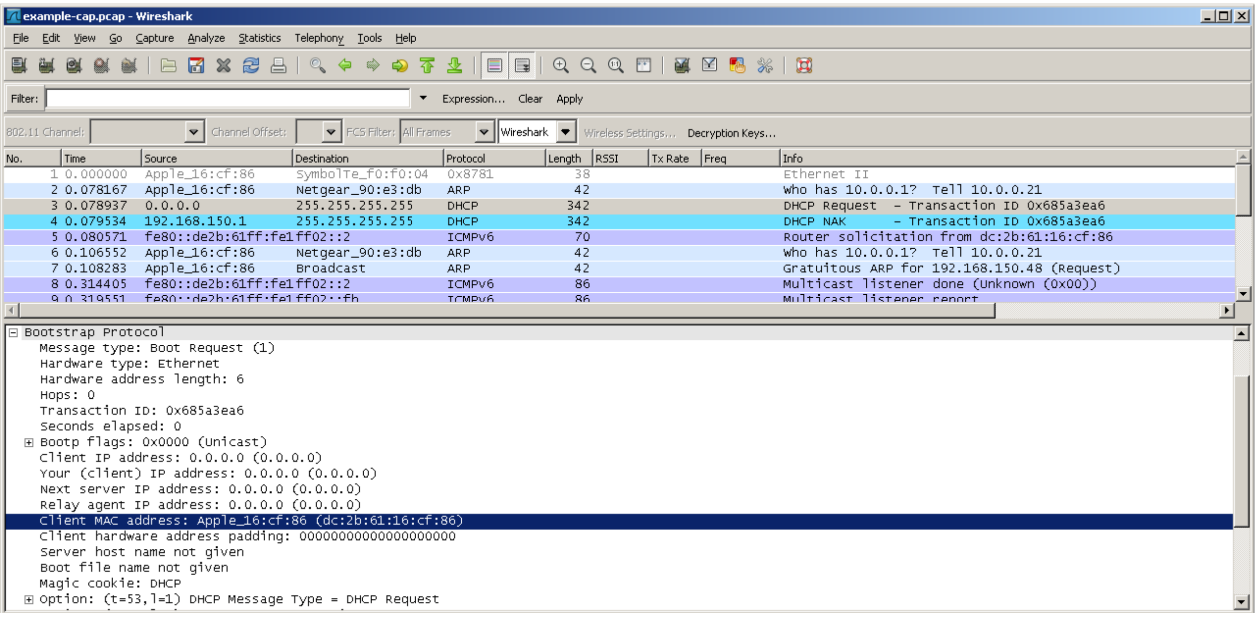
```
rfs4000-22D26E#service pktcap on inter ge1 write example-cap.pcap count 20
filter ?
  LINE  User defined packet capture filter (enclose in " if ( and ) are used):
          ([not]|)
          ether (host AA-BB-CC-DD-EE-FF|src AA-BB-CC-DD-EE-FF|dst
        AA-BB-CC-DD-EE-FF|
                 broadcast|multicast)|
          (ip6|arp|ether proto <0-65535>)|
          vlan <1-4095>|
            priority <0-7>| wlan
            <1-2>| (host|src|dst)
            A.B.C.D|
            (src|dst|) net A.B.C.D/M|
            ip (multicast|proto (<0-255>|PROTO)|)|
            tcp (fin|syn|rst|ack|)|
            udp|
            icmp|
            igmp|
            (src|dst|) port (<0-65535>|PORT)|
            capwap (data|ctrl|)|
            mint|
            radio <index>|
            dot11 (data|mgmt|ctl|beacons|probes)|
            dot11 stype <num>|
            dot11 addr <num> AA-BB-CC-DD-EE-FF|
            dot11 bss AA-BB-CC-DD-EE-FF|
            (l2|l3|l4) u8 at <0-127> value <0-255> (mask <0-255>|)| (l2|l3|l4)
            u16 at <0-126> value <0-65535> (mask <0-65535>|)| (l2|l3|l4) u32 at
            <0-124> value <0-4294967295> (mask <0-4294967295>
```

One might notice that some of the files do not appear to have been the result of the “filter” option in the command. Take for instance packet 3 in Figure 4, which is a DHCP request packet. If you drill down into this packet, you will find that the request did initiate with the client specified in the filter, as seen in Figure 5.


example-cap.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply
802.11 Channel:
Channel Offset:
FCS Filter: All Frames
Wireshark
Wireless Settings...
Decryption Keys...

No.	Time	Source	Destination	Protocol	Length	RSSI	Tx Rate	Freq	Info
1	0.000000	Apple_16:cf:86	SymbolTe_f0:f0:04	0x8781	38				Ethernet II
2	0.078167	Apple_16:cf:86	Netgear_90:e3:db	ARP	42				Who has 10.0.0.1? Tell 10.0.0.21
3	0.078937	0.0.0.0	255.255.255.255	DHCP	342				DHCP Request - Transaction ID 0x685a3ea6
4	0.079534	192.168.150.1	255.255.255.255	DHCP	342				DHCP NAK - Transaction ID 0x685a3ea6
5	0.080571	fe80::de2b:61ff:fe1	ff02::2	ICMPv6	70				Router solicitation from dc:2b:61:16:cf:86
6	0.106552	Apple_16:cf:86	Netgear_90:e3:db	ARP	42				Who has 10.0.0.1? Tell 10.0.0.21
7	0.108283	Apple_16:cf:86	Broadcast	ARP	42				Gratuitous ARP for 192.168.150.48 (Request)
8	0.314405	fe80::de2b:61ff:fe1	ff02::2	ICMPv6	86				Multicast listener done (Unknown (0x00))
9	0.319551	fe80::de2b:61ff:fe1	ff02::fb	ICMPv6	86				Multicast listener report
10	0.457570	fe80::de2b:61ff:fe1	ff02::fb	MDNS	96				Standard query PTR _raop._tcp.local, "QU" question
11	0.825621	fe80::de2b:61ff:fe1	ff02::fb	MDNS	147				Standard query ANY Viks-iPhone.local, "QU" question ANY V
12	1.099255	0.0.0.0	255.255.255.255	DHCP	342				DHCP Request - Transaction ID 0x685a3ea7

Figure 4: Viewing with Wireshark


example-cap.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply
802.11 Channel:
Channel Offset:
FCS Filter: All Frames
Wireshark
Wireless Settings... Decryption Keys...

No.	Time	Source	Destination	Protocol	Length	RSSI	Tx Rate	Freq	Info
1	0.000000	Apple_16:cf:86	SymbolTe_f0:f0:04	0x8781	38				Ethernet II
2	0.078167	Apple_16:cf:86	Netgear_90:e3:db	ARP	42				Who has 10.0.0.1? Tell 10.0.0.21
3	0.078937	0.0.0.0	255.255.255.255	DHCP	342				DHCP Request - Transaction ID 0x685a3ea6
4	0.079534	192.168.150.1	255.255.255.255	DHCP	342				DHCP NAK - Transaction ID 0x685a3ea6
5	0.080571	fe80::de2b:61ff:fe1	ff02::2	ICMPv6	70				Router solicitation from dc:2b:61:16:cf:86
6	0.106552	Apple_16:cf:86	Netgear_90:e3:db	ARP	42				Who has 10.0.0.1? Tell 10.0.0.21
7	0.108283	Apple_16:cf:86	Broadcast	ARP	42				Gratuitous ARP for 192.168.150.48 (Request)
8	0.314405	fe80::de2b:61ff:fe1	ff02::2	ICMPv6	86				Multicast listener done (Unknown (0x00))
9	0.319551	fe80::de2b:61ff:fe1	ff02::fb	ICMPv6	86				Multicast listener report

```
⊟ Bootstrap Protocol
    Message type: Boot Request (1)
    Hardware type: Ethernet
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0x685a3ea6
    Seconds elapsed: 0
  ⊞ Bootp flags: 0x0000 (Unicast)
    Client IP address: 0.0.0.0 (0.0.0.0)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: Apple_16:cf:86 (dc:2b:61:16:cf:86)
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: DHCP
  ⊞ Option: (t=53,l=1) DHCP Message Type = DHCP Request
```

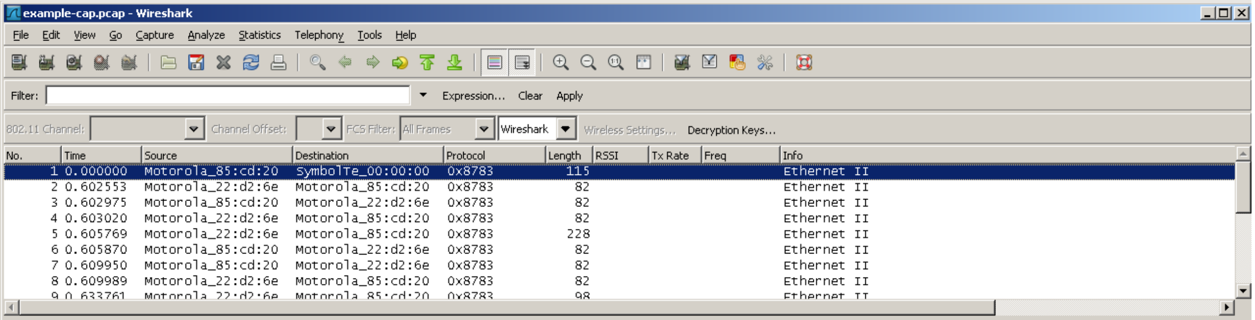
Figure 5: Filter Example

An administrator may find themselves in a position where they are troubleshooting communications between an access point and the controller. In this case one may want to start with the "capwap" filter. Control and Provisioning of Wireless Access Points (capwap) is a protocol that enables a controller to manage a collection of access points and is an IEEE standard based on Cisco's LWAPP. If these packets are not seen, then the suspect access point is not adopted (which is probably already known at this point) and further investigation into reasons why (such as layer-2 vlan boundaries) can be conducted.

Excerpt 9: Capwap Filter Example

1) Capwap Filter Example

```
rfs4000-22D26E#service pktcap on inter ge1 write example-cap.pcap count 20
filter capwap
Capturing up to 20 packets. Use Ctrl-C to abort.
20
```


example-cap.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply
802.11 Channel:
Channel Offset:
FCS Filter: All Frames
Wireshark
Wireless Settings... Decryption Keys...

No.	Time	Source	Destination	Protocol	Length	RSSI	Tx Rate	Freq	Info
1	0.000000	Motorola_85:cd:20	SymbolTe_00:00:00	0x8783	115				Ethernet II
2	0.602553	Motorola_22:d2:6e	Motorola_85:cd:20	0x8783	82				Ethernet II
3	0.602975	Motorola_85:cd:20	Motorola_22:d2:6e	0x8783	82				Ethernet II
4	0.603020	Motorola_22:d2:6e	Motorola_85:cd:20	0x8783	82				Ethernet II
5	0.605769	Motorola_22:d2:6e	Motorola_85:cd:20	0x8783	228				Ethernet II
6	0.605870	Motorola_85:cd:20	Motorola_22:d2:6e	0x8783	82				Ethernet II
7	0.609950	Motorola_85:cd:20	Motorola_22:d2:6e	0x8783	82				Ethernet II
8	0.609989	Motorola_22:d2:6e	Motorola_85:cd:20	0x8783	82				Ethernet II
9	0.633761	Motorola_22:d2:6e	Motorola_85:cd:20	0x8783	98				Ethernet II

Figure 6: Capwap capture

5.2 Example 2 – “snap”:

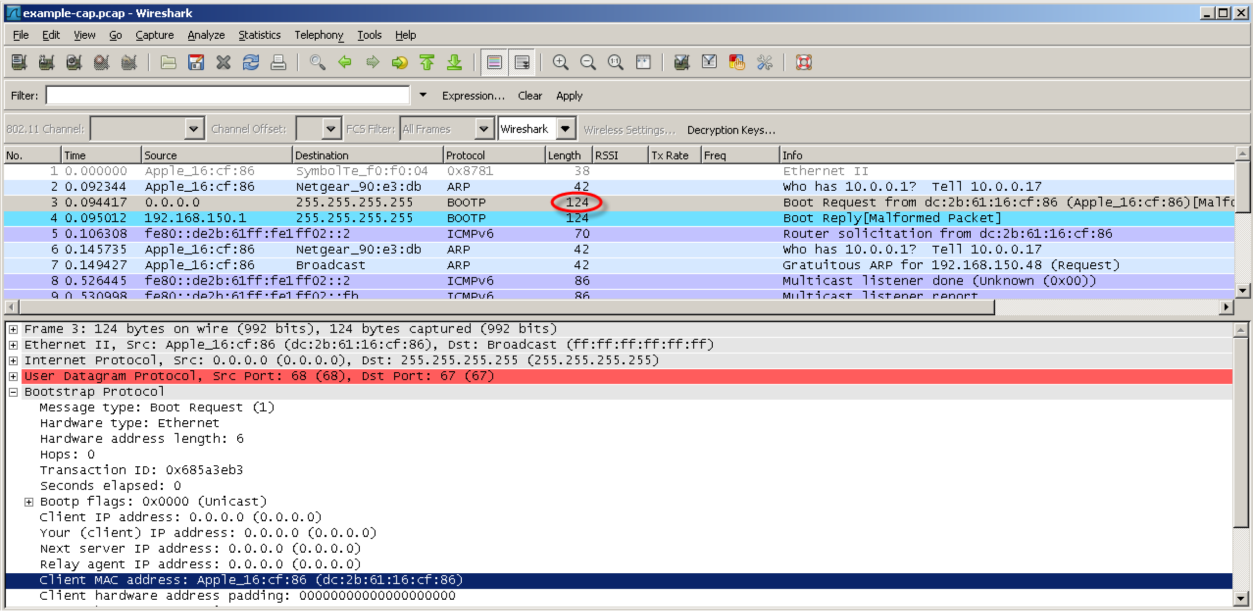
Often when troubleshooting wireless, it is not necessary for an administrator to see an entire packet’s contents; perhaps only a connectivity problem is being investigated. In this case, the “snap” option can be used to limit the size of the packets to, as an example, only the first 128 bytes. This gives the administrator the important header information that may be needed, while saving on resources such as cpu cycles, memory or storage use.

Excerpt 10: "Snap" Feature

1) “Snap” Feature

```
rfs4000-22D26E#service pktcap on inter ge1 write example-cap.pcap count 20
snap 128 filter ether host DC-2B-61-16-CF-86
Capturing up to 20 packets. Use Ctrl-C to abort.
20
```

Now compare the screen shot in Figure 6 below to the one in Figure 5. Notice that packet 3 this time is only 128 bytes in length (instead of 324 bytes) and as not all information was in the packet, Wireshark was unable to fully qualify the packet as a DHCP request. Yet the pertinent information of the “filter” client address and the BOOTP details give enough information to analyze the packet correctly.



No.	Time	Source	Destination	Protocol	Length	RSSI	Tx Rate	Freq	Info
1	0.000000	Apple_16:cf:86	SymbolTe_f0:f0:04	0x8781	38				Ethernet II
2	0.092344	Apple_16:cf:86	Netgear_90:e3:db	ARP	42				Who has 10.0.0.1? Tell 10.0.0.17
3	0.094417	0.0.0.0	255.255.255.255	BOOTP	124				Boot Request from dc:2b:61:16:cf:86 (Apple_16:cf:86)[Malfo
4	0.095012	192.168.150.1	255.255.255.255	BOOTP	124				Boot Reply[Malformed Packet]
5	0.106308	fe80::de2b:61ff:fe1	ff02::2	ICMPv6	70				Router solicitation from dc:2b:61:16:cf:86
6	0.145735	Apple_16:cf:86	Netgear_90:e3:db	ARP	42				Who has 10.0.0.1? Tell 10.0.0.17
7	0.149427	Apple_16:cf:86	Broadcast	ARP	42				Gratuitous ARP for 192.168.150.48 (Request)
8	0.526445	fe80::de2b:61ff:fe1	ff02::2	ICMPv6	86				Multicast listener done (Unknown (0x00))
9	0.530998	fe80::de2b:61ff:fe1	ff02::fb	ICMPv6	86				Multicast listener report

```
⊞ Frame 3: 124 bytes on wire (992 bits), 124 bytes captured (992 bits)
⊞ Ethernet II, Src: Apple_16:cf:86 (dc:2b:61:16:cf:86), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
⊞ Internet Protocol, Src: 0.0.0.0 (0.0.0.0), Dst: 255.255.255.255 (255.255.255.255)
⊞ User Datagram Protocol, Src Port: 68 (68), Dst Port: 67 (67)
⊟ Bootstrap Protocol
    Message type: Boot Request (1)
    Hardware type: Ethernet
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0x685a3eb3
    Seconds elapsed: 0
  ⊞ Bootp flags: 0x0000 (Unicast)
    Client IP address: 0.0.0.0 (0.0.0.0)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: Apple_16:cf:86 (dc:2b:61:16:cf:86)
    Client hardware address padding: 00000000000000000000
```

Figure 7: Example of a "snapped" capture

6. Advanced Capabilities – remote-debug:

One of the latest and most advanced features in WiNG 5 is the remote-debug command. This command allows one to troubleshoot a remote device by providing access from the controller to obtain logs, trace files and various debugging information. While this command has many more options that what are mentioned here, the focus of this writing is the **“live-capture”** option. This feature enables a WiNG 5 wireless system to function as a distributed sniffer, enabling packet capturing from remote hosts. It is a very power troubleshooting tool for the wireless and the wired network. Typically, remote-debug will be performed at a controller to capture data for known connected and specified devices (adopted AP's, cluster members, etc.).

The "remote-debug live-capture" command allows an administrator to perform packet capture of all traffic heard by the device where the remote-debug is executed, in real-time. It can then be written off to various destinations, just as a standard capture with the "pktcap" command (FTP, TFTP, file on flash, etc.). However, one big benefit is the ability to send to a remote host via the Tazman Sniffer Protocol (tzsp), allowing for real-time packet analysis with a protocol analyzer such as Omnipeek or Wireshark. The TZSP method is covered in this document.

First, we'll look at a breakdown of the command syntax and some options.

Excerpt 11: remote-debug options

1) Remote-debug command options

```
rfs4000-22D26E#remote-debug ?
  clear-crashinfo   Clear all crashinfo files
  copy-crashinfo    Copy all files from /flash/crashinfo
  copy-techsupport  Copy extensive system information useful to technical
                    support for troubleshooting a problem
  end-session       End ongoing debug session
  live-pktcap       Live packet capturex
  more              Display the contents of a file
  offline-pktcap    Capture packets and transfer packet capture data after
                    capture completes
  wireless          Wireless debug messages
```

2) Remote-debug live-pktcap

```
rfs4000-22D26E#remote-debug live-pktcap ?
  hosts      Remote hosts
  rf-domain  Specify the RF-Domain
```

When performing a live packet capture using the "remote-debug" command, one has two options; "hosts" and "rf-domain".

> *Hosts* – specify any known and reachable hosts; may specify multiple hosts by separating with spaces
>
> *Rf-domain* – specify the rf-domain relative to the data that is to be captured. This is a shortcut to listing all hosts in an rf-domain, instead of having to list each host individually.

Excerpt 12: live-pktcap "hosts" option

1 **R**

```
rfs4000-22D26E#remote-debug live-pktcap hosts ap7131-970408 ap7131-9313CC
radio 1
Capturing up to 50 packets from each remote host. Use Ctrl-C to abort
[ap7131-970408] 1 15:33:57.602665 I  BEACON  Src:C0-3F-0E-90-E3-DB Dst:FF-FF-FF-FF-FF-FF Bss:C0-
3F-0E-90-E3-DB
```

```
[ap7131-970408] 2 15:34:00.981699 I  BEACON  Src:C0-3F-0E-90-E3-DB Dst:FF-FF-FF-FF-FF-FF Bss:C0-
3F-0E-90-E3-DB   offline-pktcap    Capture packets and transfer packet capture data after
```

When capturing from multiple hosts, the controller will automatically collate the packets into a single stream in sequence, making it easy to observe events across the network.

6.1 TZSP Caveat:

TZSP is an encapsulation protocol that runs over UDP. As such, when capturing in this way, the device performing the capture will send to the specified TZSP host (a laptop running Wireshark, for instance) on UDP port 37008. As most laptops do not typically listen for udp on port 37008, a display filter can be written to hide the resulting ICMP “destination port unreachable” messages that will be received. Another option is to run the “iperf.exe” application on said host and configure it to listen on udp port 37008, but this is not necessary. “iperf.exe” can be downloaded from the Internet for free for many OS platforms by doing a simple Google search. A Windows version can be found at: http://www.noc.ucf.edu/Tools/Iperf/iperf.exe; copy this file into a system path directory, such as C:\Windows\system32.

6.1.1 CLI Configuration Example

1) Iperf.exe Command on Laptop (command prompt)

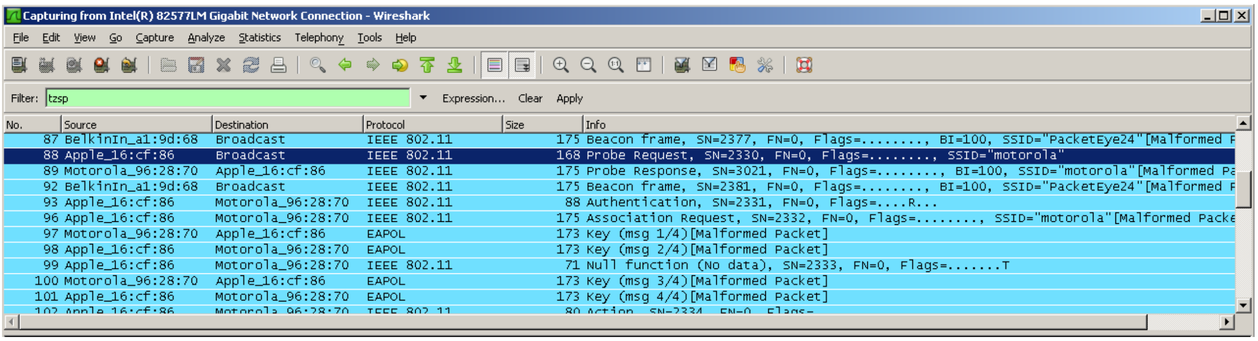
```
AP7131# iperf.exe –s –u –p 37008
------------------------------------------------------------
Server listening on UDP port 37008
Receiving 1470 byte datagrams
UDP buffer size: 8.00 KByte (default)
------------------------------------------------------------
```

2) WiNG 5 Device Command

```
AP7131# remote-debug live-pktcap rf-domain default write tzsp 192.168.150.1
radio 1
```

6.1.2 TZSP Host Configuration:

By initiating a capture on the interface as specified via IP address in the “remote-debug” command and applying a display filter of “tzsp”, real-time analysis can be performed remotely.

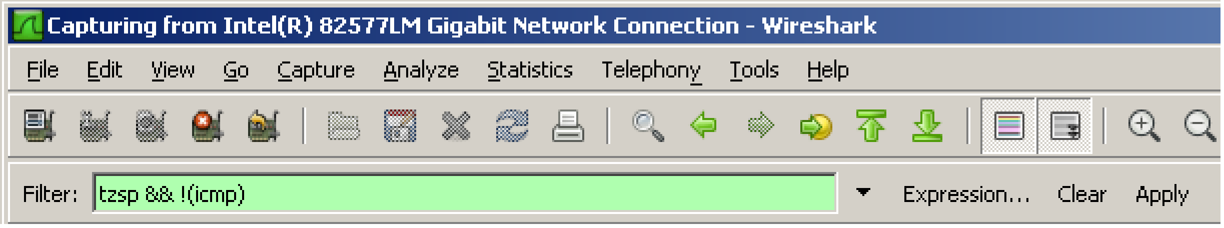

Capturing from Intel(R) 82577LM Gigabit Network Connection - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Expression... Clear Apply

Filter: tzsp

No.	Source	Destination	Protocol	Size	Info
87	BelkinIn_a1:9d:68	Broadcast	IEEE 802.11	175	Beacon frame, SN=2377, FN=0, Flags=........, BI=100, SSID="PacketEye24"[Malformed F
88	Apple_16:cf:86	Broadcast	IEEE 802.11	168	Probe Request, SN=2330, FN=0, Flags=........, SSID="motorola"
89	Motorola_96:28:70	Apple_16:cf:86	IEEE 802.11	175	Probe Response, SN=3021, FN=0, Flags=........, BI=100, SSID="motorola"[Malformed Pa
92	BelkinIn_a1:9d:68	Broadcast	IEEE 802.11	175	Beacon frame, SN=2381, FN=0, Flags=........, BI=100, SSID="PacketEye24"[Malformed F
93	Apple_16:cf:86	Motorola_96:28:70	IEEE 802.11	88	Authentication, SN=2331, FN=0, Flags=....R...
96	Apple_16:cf:86	Motorola_96:28:70	IEEE 802.11	175	Association Request, SN=2332, FN=0, Flags=........, SSID="motorola"[Malformed Packe
97	Motorola_96:28:70	Apple_16:cf:86	EAPOL	173	Key (msg 1/4)[Malformed Packet]
98	Apple_16:cf:86	Motorola_96:28:70	EAPOL	173	Key (msg 2/4)[Malformed Packet]
99	Apple_16:cf:86	Motorola_96:28:70	IEEE 802.11	71	Null function (No data), SN=2333, FN=0, Flags=.......T
100	Motorola_96:28:70	Apple_16:cf:86	EAPOL	173	Key (msg 3/4)[Malformed Packet]
101	Apple_16:cf:86	Motorola_96:28:70	EAPOL	173	Key (msg 4/4)[Malformed Packet]
102	Apple_16:cf:86	Motorola_96:28:70	IEEE 802.11	80	Action, SN=2334, FN=0, Flags=

Figure 8: Wireshark on remote host

As previously stated, TZSP encapsulates over UDP and WiNG will default to a destination port of 37008. Since it is not common that a laptop may be listening on this port, a display filter can be written to hide the ICMP “destination port unreachable” messages that will result.


Capturing from Intel(R) 82577LM Gigabit Network Connection - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Expression... Clear Apply

Filter: tzsp && !(icmp)

Figure 9: Wireshark display filters

7. Reference Documentation:

Description	Location
Zebra RFS Series Wireless LAN Switches WiNG CLI Reference Guide	http://support.symbol.com
Zebra Wireless Services Controller CLI Reference Guide	http://support.symbol.com
Zebra Remote-Debugging Functional Specification	http://netvision.sj.symbol.com/wios/browser/archive/docs/wing5/specifications/remote_debug.pdf