



Long Distance WiFi – Optimizing timeouts to prevent loss of throughput.

This document gives an overview of how you can maximize the throughput on a long distance 802.11b wireless link. In the field we use Atheros-based equipment within a NetBSD environment (CuWinWare: <http://www.cuwireless.net/>). This is a work in progress, and we don't fully understand everything the manufacturers are doing, but these techniques have helped us to bridge links up to 15km with minimal throughput loss.

When traveling over a few hundred meters, the "acktimeout" and "ctstimeout" parameters on long distance 802.11b links need some adjusting. I have found a few resources on the web that have clarified things, but nothing very official. In general WiFi cards are shipped assuming shorter distances, what 802.11b was designed for; the Atheros chipset in our D-Link DLG520 is specified, in the manual on their website:

Indoors: Up to 328 feet (100 meters)
Outdoors: Up to 1,312 feet (400 meters)

The default setting for both the acktimeout and the ctstimeout are 48usec, they max out at 744usec:

DEFAULTS on Atheros

```
# sysctl hw.ath0
hw.ath0.smoothing_rate = 95
hw.ath0.sample_rate = 10
hw.ath0.countrycode = 208
hw.ath0.regdomain = 32976
hw.ath0.debug = 0
hw.ath0.slottime = 20
hw.ath0.acktimeout = 48
hw.ath0.ctstimeout = 48
hw.ath0.softled = 0
hw.ath0.ledpin = 0
hw.ath0.ledon = 0
hw.ath0.ledidle = 270
hw.ath0.txantenna = 1
hw.ath0.rxantenna = 2
hw.ath0.diversity = 1
hw.ath0.txintrperiod = 5
hw.ath0.diag = 0
hw.ath0.tpscale = 0
```

hw.ath0.tpc = 0

48 usec is enough time for light (~300m/usec) to travel 14.4km. But the radio wave must travel from point A to point B and then back to point A in order for the acknowledgement packet to be received, so 48usec is suitable for a point-to-point link of about 7km. There would also be some additional delay from the processing of the packet, at both points.

So I'm not quite sure why the default time is so high. I can't imagine the manufacturers designing it for a 7km link by default. Maybe it has something to do with the fact that the hardware is setup for both 802.11b and 802.11g. It could be that the parameter is not directly in usec (but the general consensus on a Google inquiry says that it is). Another possibility is that 48usec is not the default, and that it is dynamically calculated on the firmware.

A link that is 30km would require a 203usec for both acktimeout and ctsttimeout. To calculate timeout from the link's distance I am using this formula:

$$\text{Timeout} = (((\text{distance in meters}) * 2) / (300\text{m/usec})) + 3\text{usec}$$

The 3usec is to give a safety margin for the processing delay. I'm not exactly sure how to calculate the processing delay; this appears to be what MadWiFi's 'athctl' program is assuming.

Changing my longest link nodes to this new value seems to significantly improve both my packet loss and my round-trip packet times:

```
# ping -s 1024 10.0.249.80
PING 10.0.249.80 (10.0.249.80): 1024 data bytes
1032 bytes from 10.0.249.80: icmp_seq=0 ttl=255 time=96.587 ms
...
1032 bytes from 10.0.249.80: icmp_seq=99 ttl=255 time=96.926 ms
^C
----10.0.249.80 PING Statistics----
100 packets transmitted, 58 packets received, 42.0% packet loss
round-trip min/avg/max/stddev = 26.698/87.117/230.151/38.130 ms
```

```
# sysctl -w hw.ath0.acktimeout=203
hw.ath0.acktimeout: 48 -> 203
# sysctl -w hw.ath0.ctsttimeout=203
hw.ath0.ctsttimeout: 48 -> 203
# ssh 10.0.249.80
root@10.0.249.80's password:
Last login: Mon Jan 30 05:02:47 2006 from 10.0.60.154
NetBSD 3.99.11 (cuw_pc) #3: Tue Nov 29 02:10:12 CST 2005
```

Welcome to NetBSD!

```
CUWIN release 0.5.9+ Unofficial, built:20051129 rev:3658
Terminal type is xterm.
We recommend creating a non-root account and using su(1) for root access.
# sysctl -w hw.ath0.acktimeout=203
hw.ath0.acktimeout: 48 -> 203
# sysctl -w hw.ath0.ctsttimeout=203
hw.ath0.ctsttimeout: 48 -> 203
# exit
Connection to 10.0.249.80 closed.

# ping -s 1024 10.0.249.80
PING 10.0.249.80 (10.0.249.80): 1024 data bytes
1032 bytes from 10.0.249.80: icmp_seq=0 ttl=255 time=117.322 ms
...
1032 bytes from 10.0.249.80: icmp_seq=100 ttl=255 time=37.612 ms
^C
----10.0.249.80 PING Statistics----
101 packets transmitted, 84 packets received, 16.8% packet loss
round-trip min/avg/max/stddev = 18.563/62.968/125.982/27.478 ms
#
```

On a different ~7km link we changed the defaults to 100 usec. Doing a bandwidth test across the link with iperf we noticed and increase in the throughput from ~15 kbytes/s to ~550 kbytes/s. VoIP went from very shaky to usable.

The following pages seem generally compliant with what we are experiencing:

<http://nuke.freenet-antennas.com/modules.php?name=News&file=article&sid=22>
http://socialfreenet.org/pipermail/discuss_socialfreenet.org/2005-January/000010.html
http://www.mikrotik.com/Documentation/manual_2.7/Interface/Wireless.html

In the end though, theory is a nice guess, but can be difficult when using newer technologies with scarce documentation. It will give you a range of parameters though; a starting point. Testing different timeout parameters, live on each link with a throughput tool, is the only way to be sure you are actually optimizing a connection.

Regards,

John Atkinson
Director - Wireless Ghana
john.atkinson@gmail.com