Wireless Means Radio

DAVID LANG

David Lang is a staff IT engineer at Intuit, where he has spent more than a decade working in the Security Department for the Banking Division. He was introduced to Linux in 1993 and has been making his living with Linux since 1996. He is an Amateur Extra Class radio operator and served on the communications staff of the Civil Air Patrol California Wing, where his duties included managing the statewide digital wireless network. He was awarded the 2012 Chuck Yerkes award for his participation on various open source mailing lists.
david@lang.hm

Wireless means radio, and it sounds obvious when it's stated, but sysadmins are normally not trained in the radio field. If the radio side doesn't work, you have no chance of the network working. In this article, I explain the problems you are facing, so that it's possible to build a reliable wireless network that will support hundreds to thousands of people using cheap commodity equipment.

Why do conference and school wireless networks always work so poorly? As IT professionals we are used to the network layer "just working" and fixing things by changing configuration files. This mind-set, combined with obvious but wrong choices in laying out a wireless network, frequently leads to a network that works just fine when tested with a small number of users, but that then becomes unusable when the crowds of users arrive. This is at its worst at technical conferences, where there are so many people, each carrying several devices, all trying to use the network at the same time, and in schools where you pack students close together and then try to have them all use their computers simultaneously.

Is this a fundamental limitation of wireless? While it is true that there are some issues that cannot be solved, there are a lot of things that the network administrator can do to make the network work better.

I have been running the wireless network for the Southern California Linux Expo (SCALE) since 2010, and this article is based on the results of the past five years' worth of SCALE conferences and the resulting paper that I presented at LISA in 2012 [1]. At the 2012 SCALE, we had 1965 attendees with 1935 unique Mac addresses on the network and 875 devices connected at peak.

The key thing to recognize when building a wireless network is that the network is primarily radios, and only secondarily digital. This doesn't mean that getting the radio side of things right will guarantee that your network will work, but it does mean that getting it wrong will guarantee that your network will not work.

The Problems

The 2.4 GHz band (b/g) has 11 channels assigned in the US, but they overlap and, as a result, you can only use three of the channels at once without problems. Three channels are really not enough as you want to leave a channel "unused" for a substantial distance between each area that is used. The rule of thumb is that if you plan to have an access point cover an area with a radius of 50 ft, you don't want to have another access point using the same channel within 200 ft. With only three channels, you can't even do this in two dimensions, let alone three, and will have to have the access points much closer together.

The 2.4 GHz band is also used extensively by other equipment, including cordless phones, cordless microphones, Bluetooth, and even microwave ovens. While the 802.11 protocol is designed to be resistant to interference from these things, they can cause packets to be corrupted, which results in retries.

Most mobile radio services suffer from the "hidden transmitter" problem. In simple terms, this is where you have three stations in a line: the station in the middle can hear stations on

each side, but the stations on the outside cannot hear each other. This prevents the stations on each end from avoiding transmitting when the one on the other end is already transmitting. When both sides transmit at the same time, the receiving station in the middle gets confused and can't make out either signal, causing both to have to retransmit the packet. In voice communication this is annoying; in digital communication, this causes everything transmitted by both stations to be garbled and both stations will have to retransmit their data.

Excessive power levels can add to the hidden transmitter problem. It is common to think that if you can't get through, turn up the power, but if only one side turns up the power, it seldom improves communications. This is because wireless networks are two-way conversations; if only one side gets louder it doesn't increase the range in which the conversation can take place but, instead, causes the stronger signal to go further and interfere with other stations.

The WiFi protocols have evolved over time, with new modes being created that squeeze more data into a given amount of airtime. In most cases the newer, higher speed modes are more sensitive to interference, so the protocol includes fallbacks to slower modes when the data is not getting through. If the problem is outside interference, weak signal and similar problems, this works very well, but if the problem is an overload of the available airtime, the result is that each station transmitting takes longer to send its signal, which makes it more likely that a hidden transmitter or other interference will corrupt the packet, resulting in retries.

802.11 has a fair amount of housekeeping traffic to let all stations in the area know that they exist and to maintain the connection to the access point. This traffic eats away at the time available and is frequently required by the spec to be transmitted at the lowest supported speed [2].

802.11n can be a benefit or a problem. The fact that it can transmit more data in a given amount of airtime can reduce congestion, but enabling the high bandwidth (dual channel) mode will require that two adjacent channels be allocated to it. Also, if the equipment is configured to operate in pure n mode, the b/g equipment will not recognize that there is a station transmitting and so will go ahead and attempt to transmit their packet.

Inappropriate use of high-gain antennas can be a problem as well. Unlike turning up the transmitter power, improved antennas help to both transmit and receive the signal. But if they are used incorrectly they will cause the station using them, in covering a larger area, to interfere with, and be interfered with by, more stations.

Mesh networks (access points connected to other access points via wireless links) require that the packets be transmitted over the radio more times, and as a result are almost always the wrong thing to use in a high-density environment.

Multi-radio enterprise access points seldom help and frequently hurt because there are already not enough channels to avoid overlapping coverage. They create large amounts of bandwidth through a single access point but decrease the overall system bandwidth by causing more interference with other access points.

Retries can also be caused by problems on the digital side of things.

The bufferbloat phenomenon [3], where the delays in getting packets to their destination can result in the packets timing out before they arrive, can also result in packets being retransmitted.

The typical collapse of wireless networks results from the combination of:

- Retries (frequently due to hidden transmitters or other interference)
- Fallback to slow speeds
- Wasted packets (due to bufferbloat and other problems)

The collapse isn't gradual; it's a performance cliff. Things work fairly well with minor slowing until you run out of airtime. At that point devices are retransmitting their data and transmitting slower (and therefore lowering the available bandwidth), and almost all useful communication just stops. If you are monitoring the network, everything will look just fine, like you are transmitting a lot of data, well below design capacity (and well below the levels you were running prior to the collapse).

The Solutions

The solution is to get as many access points into the area as you can without causing interference. To do this, you want to have each access point cover as *small* an area as possible.

First, you need to know what you are up against. Do a site survey to find out what the situation is.

- Where are the network and power jacks? I've had cases where they were eight feet apart.
- What other WiFi signals are in the area, and what channels are they on? Good tools to use are WiFi analyzer on Android or Kismet on a laptop.
- What interference is there in the area (usually not as critical as looking for WiFi signals)? My-Spy spectrum analyzer can see all signals, not just WiFi signals.
- What effect do the walls have on your signal? Movable partitions tend to block the signal more than traditional walls due to the metal mesh in the partitions.

Here are some suggestions:

Bring an AP that you can plug in and then find out where you can hear it.

Encourage the use of 5 GHz channels. There are far more of them so you can have more radios covering a given amount of floor space without interference, resulting in more bandwidth per user. In 2012 at SCALE only approximately 20% of devices were using 5 GHz, even though it had three times the capacity available.

Turn power down on 2.4 GHz to allow for more access points without overlapping footprints.

How low? At SCALE in 2012 we had the APs set to 4 mw output.

Take advantage of things that block the signal for you. In addition to walls (see above), make use of the human body, which is mostly water, which absorbs 2.4 GHz signals. Put access points low so that the crowds will prevent their signal from going as far as they normally would.

Use advanced antennas carefully. They can help you cover an area that doesn't have power or network for a fixed AP, and can help prevent interference with other areas.

Digital Issues

You should use one SSID for each band (e.g., SCALE24, SCALE5). Using a different SSID on each AP lets advanced users select the best AP for them, but it prevents roaming to a closer/better AP; if users move around they are going to have to switch SSIDs frequently. One SSID per band allows users to select the band to operate on, but then let their device use the closest AP.

Run DHCP on a central server. This similarly allows access points to act as bridges for mobile devices to roam from one AP to another without having to get new IP addresses.

Enable wireless isolation. Unless you really need the mobile devices to talk to each other, this would avoid IP-level broadcasts' many retransmissions on wireless networks.

Lengthen the beacon interval. This reduces the amount of housekeeping traffic, while lengthening the time it takes for devices to learn that the network is there or notice new APs as they move. Changing this from fractions of seconds to seconds is unlikely to cause any real problems.

Disable slow speeds. If you can disable the 802.11b speeds entirely, you avoid a significant amount of overhead. There are very few devices today that don't support at least 802.11g. If you can control what devices are in use and make sure they are all 802.11n capable, you can disable 802.11g as well.

Use APs that allow you to replace the default firmware with a Linux-based firmware that you can really configure. In 2011 we used DD-WRT on the access points, but found that it did not give us the control that we wanted, and in 2012 we used OpenWRT and were happy with it.

Disable connection tracking. Connection tracking can be a very significant overhead on the CPU and RAM of the AP. Connection tracking also doesn't work when an established connection migrates to a different AP, so it's both expensive and ineffective. Disabling connection tracking may require recompiling the kernel, but is well worth it.

If possible, disable all firewalling on the AP. Do that work upstream in order to leave as much CPU and memory available for processing packets (including doing encryption if enabled).

Set short inactivity timers. You don't want APs spending resources trying to track devices that have moved or been turned off.

Adjust kernel network buffers. The Linux wireless stack includes quite a bit of buffering inside it, so setting the kernel buffers for the wireless interfaces very low helps minimize the possibility of excessive latency. There is some recent work in this area, but it does not yet deal with the buffers inside the wireless stack [4].

Set up monitoring. If you don't know what's going on you can't fix it.

One year we had a serious problem with people turning off the power on access points or unplugging them. Without monitoring you won't know when something goes wrong. An AP that is still powered but not on the network is far worse than one that is completely dead, as user devices keep trying to connect to it.

With hundreds to thousands of users, you will never have enough Internet bandwidth to satisfy everyone. So you should implement normal site bandwidth-saving tools such as blocking streaming sources, and implement QoS traffic shaping to provide fairness between users.

Additionally, packet timeouts and bufferbloat latencies are more likely the most hops any one network connection has, so you can avoid a lot of problems if you can have the users connecting to a local machine that then acts as a proxy. Running a Web caching proxy server or a local mirror for popular distro repositories both saves you Internet bandwidth and changes the user connections from long distance to local connections. The number of people who opt to do system updates at large events is surprising.

Once you understand what problems you are facing on the radio side of the equipment, you can plan accordingly and build a reliable wireless network that will support lots of users, and do it using commodity equipment.

References

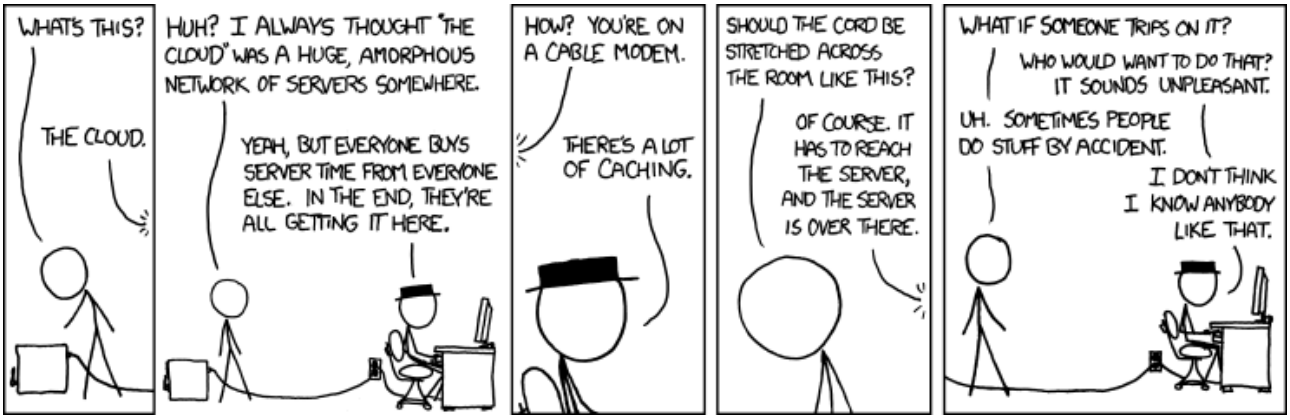
[1] https://www.usenix.org/conference/lisa12/building-wireless-network-high-density-users.

[2] Any broadcast traffic (such as SSID broadcasts, connection requests, etc.) must be transmitted at the lowest speed supported so that devices that only support that speed and not higher ones will still be able to decode the message.

[3] In an attempt to prevent packet loss, and with memory becoming vastly cheaper over time, buffers on network devices have become very large. If there is significant congestion and the buffers stay full for an extended time, packets can sit in the buffers so long that by the time they arrive at their destination they have already timed out and a replacement packet has been sent.

[4] http://www.bufferbloat.net; https://lists.bufferbloat.net/listinfo/cerowrt-devel.

xkcd



xkcd.com