This is an obsolete Draft.

The frozen, final draft is at:

new_edit_fcc_alternate_guidelines-final.odt

Comment Signatures now near end of document...

DRAFT AS YET. Not for publication!*

Before the

FEDERAL COMMUNICATIONS COMMISSION

Washington, DC 20554

In the Matter of	)	
	)	
Amendment of Part 0, 1, 2, 15 and 18 of the	)	ET Docket No. 15-170
Commission's Rules regarding Authorization	)	
Of Radio frequency Equipment	)	
	)	
Request for the Allowance of Optional	)	RM-11673
Electronic Labeling for Wireless Devices	)	

Summary

The rules laid out in ET Docket #15-170 should not go into effect as written. They would cause more harm than good and risk a significant overreach of the Commission's authority. Specifically, the rules limit the ability to upgrade or replace the firmware in commercial, off-the-shelf home or small-business routers. This would damage the security, reliability and functionality of home and business networks. It would also restrict innovation and research into new networking technologies.

We present an alternate proposal that better meets the goals of the FCC - not only ensuring that the desired operation of the RF portion of a Wi-Fi router is within the mandated parameters, but also assisting the FCC's broader goals of increasing consumer choice, fostering competition, protecting infrastructure and increasing resiliency to communications disruptions.

As experts in modern software engineering and networking, we recommend the FCC mandate best practices for software development as described in "An Alternate Approach". In short, that the vendor will:

- Provide full and maintained source code for review and improvement.
- Assure that secure firmware updates are available and under owner control.
- Address known security vulnerabilities in source and binary within specific time frames.
- Be made aware that noncompliance could result in decertification.

We also ask that the FCC review and rescind existing rules that conflict with these best practices.

Implementing the above rules will increase the security, reliability and functionality of home and business networks by ensuring that software and hardware be allowed to evolve in response to new bugs, threats and regulations.

If the Commission does ***not*** intend to prohibit the upgrade or replacement of firmware in Wi-Fi devices, the undersigned would welcome a clear statement of that intent. Furthermore, we would welcome rules that actually encourage equipment vendors to make their software inspectable and repairable.

Introduction

The undersigned are experts in modern software engineering and networking technologies. We are deeply familiar with modern best practices for software development, network deployment, and Internet security.

The proposed rulemaking addresses the concern for modular transmitters for licensed uses. However, as currently proposed, these rules would cause much more harm than good. While the rules might protect portions of the radio spectrum against non-compliant consumer routers, the proposed implementation would also forbid vital work that is in the public interest. There is also a substantial risk that the ruling will lock in place software that is dangerously buggy and insecure.

Not only are Wi-Fi routers an area of rapid innovation, they also represent a key vulnerability in the security of home and small business networks. However, the economics of home routers are such that vendors lack any incentives to do substantial work in the area of securing home routers, nor adding IPv6 support, nor improving home router performance in the presence of congestion, because there is no market feedback that allows vendors with better products to charge higher prices or offer ongoing maintenance.

As a result of this, the academic community, the free and open source software (FOSS) community, and the Internet Engineering Task Force (IETF) have been forced to take up the slack. We have been actively working to resolve problems in this area for over four years, and have been releasing our standards-conformant implementations to the public using the very mechanisms that the proposed rulemaking might prohibit.

Because this is an area of interest to academics, open source developers and the IETF, the work coming from those communities is in turn being adopted by some forward-looking Wi-Fi vendors in their new products. This proof-of-concept work simply would not be possible without the ability to flash firmware on existing and upcoming commodity Wi-Fi routers.

We are most familiar with improvements in internet security, IPv6 capability, correctness, and performance delivered via the “WRT” family of open-source router operating systems: OpenWrt, DD-WRT, CeroWrt and derivatives. The proposed rulemaking asks that router vendors “describe in detail how the device is protected from flashing and the installation of third-party firmware such as DD-WRT.” This is exactly backwards; the options provided by projects like OpenWrt are the only realistic hope of addressing severe problems with the present network.

In the note that follows, we describe our concerns with present-day Wi-Fi routers, the work of the IETF and FOSS communities, the relevance of those efforts to the protection of the radio spectrum, and suggest at least one way to resolve those problems using standards-compliant software engineering best practices.

Our Concerns with Current-day Wi-Fi Routers

The United States has hundreds of millions of Commercial Off-The-Shelf (COTS) routers installed in homes, small businesses, and enterprises. Although these routers have all passed FCC certification for their RF operation, all current implementations have significant and severe software flaws.

Insecure Implementations in the Router

These routers generally ship with outdated kernel software, frequently four or more years out of date. Too often, the firmware is insecure out of the box at FCC certification time. As examples of this, an evaluation by Independent Security Evaluators (ISE)[1] demonstrated multiple vulnerabilities in common routers, many of which can be exploited without active user participation. Other security researchers present similar results.[2] More recently (August 2015), a router vendor was found to have a critical vulnerability that could compromise user data or allow an attacker complete control of the device.[3]

The economics of the present day home router market makes this unlikely to change,[4] [5] which makes the ability to install corrected, open, firmware essential for ensuring the future safety and integrity of the internet on existing and future devices.

Lack of Functionality

Most Wi-Fi routers barely support IPv6, if they implement it at all. There are many poor implementations of IPv6, leading to problems with interoperability[6]. Few vendors are tracking the

---

[1] https://securityevaluators.com/knowledge/case_studies/routers/soho_router_hacks.php and https://securityevaluators.com/knowledge/case_studies/routers/soho_service_hacks.php
[2] https://www.defcon.org/images/defcon-18/dc-18-presentations/Heffner/DEFCON-18-Heffner-Routers.pdf
[3] Belkin N600 DB Wireless Dual Band N+ router contains multiple vulnerabilities http://www.kb.cert.org/vuls/id/201168
[4] https://cyber.law.harvard.edu/events/luncheon/2014/06/gettys
[5] https://www.schneier.com/essays/archives/2014/01/the_internet_of_thin.html
[6] See http://mailman.nanog.org/pipermail/nanog/2015-October/079755.html and subsequent messages

developments of the IETF working groups, and none are working on retrofitting their previously shipped products. There's no money in that retrofit, so open-source upgrades are the only way it will happen.

Furthermore we are not aware of any COTS Wi-Fi routers that correctly implement the Domain Name System SECurity (DNSSEC) specification, which provides significant protection against DNS attacks.

Inability to verify proper operation

The recent Volkswagen case shows that using closed and uninspectable source code can lead to devices (vehicles) that surreptitiously operate out of specification. Similarly, the inability to inspect the RF-controlling source code of routers makes it impossible to determine whether they are actually operating within specifications under all circumstances, not just on an FCC certification run: home routers are marketed on the basis of wifi speed; exceeding the FCC rules would be akin to violating the EPA rules.

Inability to correct/improve operation

The proposed rules would make the owner of the device unable to repair it if it is not in compliance. This directly contradicts the principle that an operator is responsible to ensure that a device operates within the regulations, even if it has a manufacturing defect.

Summary of Concerns

Chipmakers and hardware designers generally build in the ability to update the software easily, so that problems can be solved, and so that additional functionality can be added over time, not only when the machine is first produced. However, many of the companies selling the hardware have a single package of software that they load on their products until the product no longer sells, and then abandon it. They fail to update their software, even though their suppliers, the software development community, and their own customers expect it. Absent some kind of vendor liability[7] for errant software, market forces will not correct this. At present, the aftermarket firmware is the only feedback that exists.

7 see point "3. Source code liability -- CHOICE" of http://www.privacywonk.net/2014/11/cybersecurity-as-realpolitik-by-dan-geer.php

Present-day factory firmware with its outdated technology and binary blobs has been widely shown to be buggy, insecure, inefficient, and badly written. Consequently, as network engineers, we never use factory firmware on a home router if we can help it - it is too risky.

Work of the IETF, FOSS community, and the CeroWrt Team

The Internet Engineering Task Force (IETF) is the group of network professionals who developed the Internet, and whose continuing goal is "to make the Internet work better."[8] They are deeply knowledgeable about networking best practices, and the design of future capabilities.

The Free and Open Source Software (FOSS) community is a group of software experts who collaborate to produce software that is generally available at no cost, and without restrictions on its use.[9] Volunteers and paid professionals may work on FOSS projects to build good and useful software.

Groups like these often combine to perform research outside the traditional industrial or academic settings. By using free and open tools (compilers, etc.) and inexpensive equipment (Wi-Fi routers), research teams can produce new insights into network phenomena and improve the way the network operates.

We are most familiar with the OpenWrt and CeroWrt[10] efforts, but other projects (such as DD-WRT[11], Tomato[12], Gargoyle[13], and others) operate in much the same collaborative fashion.

The CeroWrt Project

The members of the CeroWrt router research project have worked over the past four years to address the problem of how to speed up the edge of the network under load. Many network connections perform quite well under the load presented by a single user, but break down when more than one user attempts to share the network connection. This project—made possible only by entirely open and

---

8 IETF: http://ietf.org/

9 FOSS: https://en.wikipedia.org/wiki/Free_and_open-source_software

10 CeroWrt Home page: http://cero2.bufferbloat.net/cerowrt/ wiki: http://www.bufferbloat.net/projects/cerowrt

11 DD-WrT Home Page: http://dd-wrt.com

12 Tomato Home Page: http://www.polarcloud.com/tomato

13 Gargoyle Router Home Page: https://www.gargoyle-router.com/

modifiable firmware—made a breakthrough in 2012, speeding up the edge of the internet often by an order of magnitude or more under load.

The problem is called “Bufferbloat”[14] - and the best known current solution, arrived at through the work of the CeroWrt project, is called “fq_codel”[15].

Bufferbloat causes bad network performance for voice, video-conferencing, gaming, DNS and web traffic. It is now easily tested for[16] and is currently at epidemic proportions across the edge of the Internet[17].

Standardization efforts for fq_codel are nearly complete within the IETF “aqm” working group and work is in progress for a successor algorithm, “cake”[18], which addresses a few edge cases that fq_codel did not.

Fq_codel has seen global adoption as a result of its inclusion in modern versions of the Linux kernel, which is used in many Wi-Fi customer-premise routers, home routers, and larger devices. It has been shown to work extremely well on wired packet transports (DSL, cable, fiber, ethernet), and can provide substantial improvements to point-to-point wireless links.

While commercialized versions of fq_codel are now appearing in multiple new router products[19], all thus far have missed a fundamental problem involving network offloads (fixed in “cake”), and are themselves in need of a firmware upgrade.

The work on CeroWrt and fq_codel was instrumental in Apple Computer’s recent decision to enable “Explicit Congestion Notification” (ECN) across their IOS and OSX operating systems, making possible loss-free, low latency, and congestion-controlled network video transfers[20].

---

14 Bufferbloat: http://www.bufferbloat.net/projects/cerowrt/wiki/Bloat-videos

15 fq_codel: https://tools.ietf.org/html/draft-ietf-aqm-fq-codel-01

16 Test for Bufferbloat at: http://dslreports.com/speedtest

17 http://www.dslreports.com/speedtest/results/bufferbloat?up=1

18 http://www.bufferbloat.net/projects/codel/wiki/CakeTechnical

19 Ubiquiti’s edgerouter series http://community.ubnt.com/t5/EdgeMAX-Updates-Blog/EdgeMAX-EdgeRouter-software-release-v1-7-0/ba-p/1287631 , “streamboost”, and otherrebranded QoS systems from netgear and others.

20 Apple Computer uses Explicit Congestion Notification: https://developer.apple.com/videos/play/wwdc2015-719/

In summary, the CeroWrt team has made an important research contribution to address Bufferbloat. The ability to flash and update custom firmware against many brands of routers has been essential to our research.

Making Wi-Fi Fast

Having conquered Bufferbloat on non-wireless transports, the core CeroWrt team has begun a project (Make-Wi-fi-Fast)[21] that will make vast improvements in how Wi-Fi functions with multiple stations in use. The solutions use open and unpatented code, and are being done in conjunction with proposed standards inside the IETF. This code can be incorporated into new Wi-Fi products, AND — more importantly — can be retrofitted into hundreds of millions of non-locked down existing products. This code is all within the existing capabilities of Wi-Fi, and does not affect regulatory compliance. Work also in progress will improve spatial reuse of the RF spectrum, and dramatically[22]reduce power usage in the general case. The latter should better support sharing of Wi-Fi spectrum in higher density situations, for example in apartment buildings and shared industrial settings, where more than one company's network infrastructure's radiation sphere overlaps. This is directly in line with the FCC's core mandate to prevent destructive interference.

Dave Täht[23] outlined some of these fixes in talks to both the IEEE 802.11 working group,[24] and – more recently, and filmed - at the BattleMesh conference.[25] We believe that the demonstration, 27 minutes in, of what is wrong with packet aggregation in 802.11n and later in present-day drivers and firmware will be a wake-up call for everyone working in this area.

Again, progress in this project will rely on being able to make frequent updates to the firmware of standard Wi-Fi routers.

Security

21 MakeWi-FiFast Design document: https://docs.google.com/document/d/1Se36svYE1Uzppe1HWnEvat_sAGghB3kE285LElJBW4/edit?usp=sharing

22 Minstrel-Blues - coupled power and rate control - https://github.com/thuehn/Minstrel-Blues

23 Dave Täht's blog: http://the-edge.blogspot.com/

24 Fixing Wi-Fi: http://snapon.lab.bufferbloat.net/~d/ieee802.11-sept-17-2014/11-14-1265-00-0wng-More-on-Bufferbloat.pdf

25 How to make Wi-Fi Fast again: https://www.patreon.com/dtaht

In addition to making the changes for bufferbloat and Making-Wi-Fi-Fast, we have made our firmware considerably more secure than what most home routers ship with today. We have incorporated multiple advanced hardening features and adopted the latest security related standards such as DNSSEC[26]. Our work adds full support for IPv6 related protocols, many measurement and diagnostic tools[27], the latest fq_codel[28] and latest "cake"[29] Bufferbloat-fighting algorithms, and the latest IETF "homenet" working group outputs – hnetd[30] and babel[31].

Best Practices

The development process we've described above has been combined into the latest OpenWrt production version. OpenWrt has been independently developed by hundreds of researchers and developers spread across the globe. Although teams collaborate on independent tasks, the central Linux and OpenWrt software combines those projects into a whole. The result is a software platform with separate pieces that can combine to support next generation IPv6 Internet, the latest, most secure versions of the protocols that run on the Internet, and that delivers performance adequate to the speeds being delivered by the most innovative service providers, such as Comcast, Google Fiber and Verizon FiOS.

Software and hardware development are iterative processes. No hardware, software or firmware is perfect on first release. Because of this, we believe that any rules that assume a one-time certification process will not and cannot be successful. Most home router software today is shipped with ancient code, rife with security holes and bugs.

All software and hardware must be able to evolve. No one-time certification can possibly be enough to meet all the goals of regulatory compliance, addressing bugs, and meet new needs and threats on the Internet.

---

26 An acronym for **D**omain **N**ame **S**ystem **SEC**urity - a means of securing DNS lookups to detect and ignore false responses from attackers

27 https://data.fcc.gov/download/measuring-broadband-america/2014/Technical-Appendix-fixed-2014.pdf

28 fq_codel internet draft: https://tools.IETF.org/html/draft-hoeiland-joergensen-aqm-fq-codel-01

29 Comprehensive queue management: http://www.bufferbloat.net/projects/codel/wiki/CakeTechnical

30 https://github.com/sbyx/hnetd/

31 http://www.pps.univ-paris-diderot.fr/~jch/software/babel/

We therefore routinely change, improve and correct errors in the main body of the software, using a policy of wide availability, peer review and regular updates, especially ones containing corrections provided by the owners of the devices, contributions from FOSS developers around the globe, and others sponsored by the IETF, in this case the aqm and homenet working groups.

The source code to this software is under full change control. This software contains built-in configurations to ensure that radios are used in compliance with the local laws and regulations.

Relevance to the Proposed Rulemaking

One of the biggest barriers to a final implementation of the Make-Wi-Fi Fast project is the FCC's latest regulatory rulings, particularly as makers of firmware have understood them.

The most common interpretation is that these rules require vendors to deny access to (or, more importantly, the ability to modify) their code to the FOSS, IETF, and academic research communities, as well as other potential investigators who do not have the purchasing power that would allow them to request special treatment.

In some cases, vendors have used FCC rulings as an excuse to keep their source code private, unmodifiable, and unfixable. This behavior locks existing bugs and security vulnerabilities in place, with potential consequences so grave that they qualify as a present danger to the national security of the U.S. Router firmware, in particular, is a natural target for both criminal exploitation and cyberwarfare. To mitigate these risks, field upgrading and open-source development are both essential.

Our understanding is that portions of the existing and proposed FCC rules[32] will prevent necessary in the field testing and fixes for router firmware to improve internet and device security, fixes to buggy code all over the software stack, notably upgrades and improvements to IPv6, and fixes to improve Wi-Fi performance and interoperability - in addition to making in the field compliance with future FCC regulations difficult.

32 http://transition.fcc.gov/Daily_Releases/Daily_Business/2015/db0722/FCC-15-92A1.pdf

The proposed rules may prevent anyone other than the original vendor from changing anything – at all. At least one vendor states that FCC compliance as a reason to require firmware activation codes,[33] and members of the community have observed several new instances of vendor firmware being locked down where it was not previously.[34] In addition, at least one chipset vendor has stated that locking down firmware would be "the easiest way to comply".[35] Together, this strongly indicates that the tendency of vendors to lock down firmware will become more prevalent as a consequence of the new rules.

The portion of the software on common Wi-Fi routers that actually affects the correct functioning of the radio is a tiny fraction of the operating system: a "device driver" and the "co-processor firmware". This is a small, separable and specialized component that operates the radio hardware. Despite its small size and specialized function, its programmatic and hardware interfaces must too, change and evolve in conjunction with and in response to other changes in that operating system and programs on the router.

A prohibition on the owner of the router replacing any part of the operating system has a chilling effect on our ability to implement new algorithms. It currently limits our attempts to fix the ath9k and mt76 devices, and is stopping us cold in attacking similar problems in the universally closed firmware in 802.11ac devices. Proposed upcoming rules may prevent further work on the project at all.

The proposed rules also would cause difficulties to the FCC's own measurement studies[36] attempting to analyze Wi-Fi behavior![37]

The FCC *should not* allow the development of equipment that cannot be inspected, as that has led to software that covertly avoids regulatory compliance, as has recently been found to be the case with Volkswagen cars with diesel engines. Source code transparency in this case would have assisted necessary regulation, and can do so for network hardware as well.

---

33 https://www.ubnt.com/fcclabelrequest/

34 See e.g. http://lists.prplfoundation.org/pipermail/fcc/2015-September/000342.html

35 http://arstechnica.com/information-technology/2015/09/fcc-open-source-router-software-is-still-legal-under-certain-conditions/

36 https://data.fcc.gov/download/measuring-broadband-america/2014/Technical-Appendix-fixed-2014.pdf

37 http://projectbismark.net/

We understand there are significant concerns about existing other users of the Wi-Fi spectrum, and a desire to avoid uncontrolled change. We advocate well-controlled change with established software engineering practices. However, we most strenuously advise against prohibiting all change to any device that contains radio components. By treating compliance-critical and non-compliance-critical software as if they were the same, the FCC would block efforts to address serious ongoing problems with the Internet infrastructure.

In our work, we have found and fixed many bugs that significantly mitigated misuses of Wi-Fi spectrum. For example, amidst hundreds of others, we found and fixed a quite egregious infinite retry bug. This bug caused the device driver to become locked and send the same data over and over again for tens of seconds, improperly dominating the channel, severely and needlessly interfering with other users.[38]

While the goal of protecting some radar installations from non-compliant equipment is important, the restrictions this rulemaking would create would have a much broader effect, potentially reaching the majority of users of the Internet, not only within the United States but also abroad. We believe that the FCC should instead focus on ways to improve how Wi-Fi and software defined radio (SDR) are developed and used.

Software Engineering Best Practices

Software and hardware development are iterative processes. No hardware, software or firmware is perfect on first release. Because of this, we believe that any rules that assume a one-time certification process will not be successful. Most home router software today is shipped with ancient code, rife with security holes and bugs. All software and hardware must be able to evolve. No one time certification can possibly be enough to meet regulatory compliance, nor address bugs, nor meet new needs or threats on the Internet.

38 http://www.bufferbloat.net/issues/216

We therefore routinely change, improve and correct errors in the main body of the software, using a process of wide availability, peer review and regular updates, especially ones containing corrections provided by the owners of the devices, contributions from FOSS developers around the globe, and others sponsored by the IETF working groups, in this case the aqm and homenet working groups.

The source code to the OpenWrt project is under full change control, with the authors of each patch publicly logged. This software contains built in configurations to ensure that radios are used in compliance with the local laws and regulations throughout the world.

We recommend that the same degree of professional source code management and change control be applied to all products submitted to the FCC, and other U.S. regulatory bodies such as the FTC, and EPA. We encourage you to apply it not just to the general software and operating system, with which we are primarily concerned, but also to the radio device drivers and on-board firmware with which you have concerns - for which we also have upgrades pending (Minstrel Blues[39])for that we cannot further develop or deploy in today's regulatory environment.

Don't Prohibit Third-Party Software

The proposed rules give guidance that suggests that vendors declare "how the device is protected from 'flashing' and the installation of third-party firmware such as DD-WRT"[40] on new wireless hardware. This is destructive on many levels.

1. The best available software to date is a derivative of the newest releases of the Linux and BSD based code. DD-WRT (an OpenWrt relative) tracks these somewhat closely, and is actually shipped as a factory default by at least one vendor (Buffalo). OpenWrt itself is shipped as the default in thousands of projects, and used by QCA, at least, as their default operating system (OS) supplied to their original design manufacturers (ODMs). Many other firmware builds are based on open source distributions such as Debian (Ubiquiti), or buildroot (google fiber). Other

39 https://github.com/thuehn/Minstrel-Blues

40 https://apps.fcc.gov/kdb/GetAttachment.html?id=1UiSJRK869RsyOddPi5hpw%3D%3D&desc=594280%20D02%20U-NII%20Device%20Security%20v01r02&tracking_number=39498

big-name vendors in the Wi-Fi market known to use a fork of OpenWrt are Meraki, and Ubiquiti, but no doubt many others exist.

2. The binary firmware for at least one 802.11ac device (QCA ath10k) is actually based on the BSD driver stack, which is another open source operating system. Despite it having the open-source BSD license, FCC concerns over needing lockdowns have been a huge factor in delaying the open, public release of that firmware. This in turn prevents it from receiving needed fixes, and the firmware is still, after 2+ years of shipping, not ETSI CCA compliant. Concerns over FCC's rules have even been making it impossible for those with source licenses to that firmware to collaborate – which includes the original authors of that BSD-based code!

3. These rules ill-advisedly attempt to regulate the means instead of the end results. Third party software is demonstrably better in many regards than most vendor's stock firmware. It is not in the public interest to prohibit its use where a much more limited regulatory approach would succeed.

An Alternate Approach

In an effort to assure regulatory power and channel usage compliance, the Linux community has long made available a secured, signed and regularly updated worldwide database[41] of regulatory power and channel constraints for all devices that use Wi-Fi. It, and code to access it, is published openly, online, using a distributed change control system called git, and is fully available for anyone with an Internet connection to inspect and use. This is at present the best mechanism for ensuring that devices—once configured properly for their locality—are compliant. Mandating that this database be used is certainly within the scope of what the FCC can demand of devices in this spectrum. In combination with unambiguous locality configuration, this can be used to ensure that the device is compliant except by willful negligence of the owner.

41 https://wireless.wiki.kernel.org/en/developers/regulatory

The portion of the source code that controls the radio is very small compared to the entirety of the underlying operating system, graphical user interface, routing and switching code, and all the other functions that make up a modern Wi-Fi router. As such, restricting the entire firmware toware carries with it a lot of collateral damage by also preventing improvements to these other parts. Instead, we propose below that the radio-specific code be placed under the same professional change management and review process as the rest of the operating system.

Recommendations

In place of these regulations, we propose that the Commission adopt rules that would foster innovation, improve security, make Wi-Fi better, and overall improve usage of the Wi-Fi spectrum for everybody.

Specifically we advocate that, rather than denying the ability to make any changes to the router whatsoever, that the rules should mandate that router vendors open up their code (especially the code that controls the RF parameters) to describe and document the safe operating bounds for the software defined radios within the Wi-Fi router.

This path has the following advantages:

- **Inspectability.** Skilled developers can verify the functioning of the software drivers that are now hidden in binary blobs.

- **Opportunity for innovation.** There are many experiments that can be performed to make the network “work better” while not affecting compliance.

- **Improved spectrum utilization.** A number of techniques to improve use of the Wi-Fi bands remain theoretical possibilities. Field trials with the proposed algorithms could prove (or disprove) their utility, and advance the science of networking.

- **Fulfillment of legal (GPL) obligations.** Allowing router vendors to publish their RF-controlling source code in compliance with the license under which they obtained it will

free them from the legal risk of being forced to cease shipping code that they no longer have a license for.

To accomplish these goals the FCC could mandate that:

1. Any vendor of a SDR, wireless, or Wi-Fi radio of any sort - in order to achieve FCC compliance - must make public the full and maintained source code for the device driver and radio firmware. The source code should be in a buildable, change controlled source code repository on the Internet, available for review and improvement by all.

2. The vendor must assure that secure update of firmware be working at shipment, and that update streams be under ultimate control of the owner of the equipment. Problems with compliance can then be fixed going forward by the person legally responsible for the router being in compliance (that is, its owner).

3. The vendor supply a continuous stream of source and binary updates, that must respond to regulatory transgressions and Common Vulnerability and Exposure reports (CVEs) within 45 days of disclosure, for the warranted lifetime of the product + 5 years after last customer ship.

4. Failure to comply with these regulations should [or could] result in FCC decertification of the existing product and in severe cases, bar new products from that vendor from being considered for certification.

In addition, we ask that the FCC review and rescind rules for anything that conflicts with open source best practices, produces unmaintainable hardware, and/or which causes the vendors to believe they should hide the mechanisms they use by shipping undocumented “binary blobs” of compiled code, or lock down mechanisms that forbid user patching. This is an ongoing problem to all interested parties in the internet community trying to do change control and error correction on safety-critical systems.

Rationale

Requiring that ALL manufactures of Wi-Fi devices make their source code publicly available and publicly maintained levels the playing field so that none can behave badly. The recent Volkswagen scandal with uninspected computer code that cheated emissions testing shows that this is a **real concern**.

Given the above best practices, any competent engineer could then assess regulatory compliance in an afternoon, a change in costs which would greatly assist the FCC in coping with the greatly increased volume of certification requests.

As individuals involved in making the edge of the Internet and Wi-Fi better, we want both to comply fully with the law and to move the state of the art forward. The nexus that will both allow us to achieve our goals and the FCC to carry out its function is transparent access to source code.

Conclusion

These present day – and pending – FCC regulations regarding the design and use of the public's Wi-Fi devices have been a significant barrier to actually making them work faster, safer, and better in the general case in the air and on the Internet itself.

A pro-transparency position on the FCC's part will make for a better future for billions of Wi-Fi devices already deployed, and the billions to come, as well as a freer, faster, safer Internet.

We therefore recommend the FCC distinguish between the licensed radio hardware and software and any other software running under the same device and require well-tested change-control and public source practices to guard against and immediately remediate problems, specifically including compliance problems.

All our measurements show that the state of Wi-Fi today is dismal[42] - and we know why. If the FCC permits us to soldier on, we will make Wi-Fi, and the Internet, a whole lot better.

Sincerely,

Dave Täht US Citizen Co-founder, bufferbloat.net	Vint Cerf US Citizen Co-Inventor of the Internet

[42] The good, the bad, and the Wi-Fi: http://www.sciencedirect.com/science/article/pii/S1389128615002479

Thanks for comments and editing help from:

Chuck R Anderson

Eric Bishop

Franz Böhm

Robert Bradley

Rich Brown

David Collier-Brown

David Hilton

Toke Høiland-Jørgensen

Randell Jesup

Erik Kline

Camden Lindsay

Bill Moffitt

Luigi Mori

Michael Richardson

Bruce Perens

Maxim Sobolev

Eric Schultz

Keith Winstein

Eric Raymond

Anonymous

calvert

cschulze85

Sascha Meinrath

Additional Signatures

are being collected via web form at:

http://goo.gl/forms/WCF7kPcFl9