The TKIP Hack

Description: Steve and Leo begin with a refresher on WEP, the original technology of WiFi encryption. With that fresh background, they then tackle the detailed explanation of every aspect of the recently revealed very clever hack against the TKIP security protocol. TKIP is the older and less secure of the two security protocols offered within the WPA and WPA2 WiFi Alliance certification standards.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-170.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-170-lq.mp3

INTRO: Netcasts you love, from people you trust. This is TWiT.

Leo Laporte: Bandwidth for Security Now! is provided by AOL Radio at AOL.com/podcasting.

This is Security Now! with Steve Gibson, Episode 170 for November 13, 2008: WPA Crack. This show is brought to you by listeners like you and your contributions. We couldn't do it without you. Thanks.

It's time for Security Now!, the show that covers all those important little gotchas on the Internet, on the computer, in your banking, in your online privacy. And who better to do that than Mr. Steve Gibson, head honcho.

Steve Gibson: Hey, Leo, great to be with you again.

Leo: Yeah, it's good to talk to you, Steve.

Steve: 170.

Leo: 170, can you believe that? The man at GRC.com, and also author of SpinRite, his fantastic disk recovery utility. Discovered the first spyware in the world and coined the name "spyware." And today we are going to change gears a little bit.

Steve: Yes. I said last week that we were going to have the author of Sandboxie on to talk about his program and the way it functions, it being a cool way of encapsulating pretty much any program, but specifically web browsers - which of course, as we know, have so much trouble with security - in order to prevent anything the web browser might do from escaping. We've talked about the problems of disabling scripting, which is a really good thing to do. But it just - so many sites are increasingly requiring scripting that it becomes a problem to disable it.

Anyway, in the meantime the big news hit, between the time I said that last week and now, about a - well, we're going to talk about this WPA WiFi security problem. Everything I have read in the press has been wrong.

Leo: Oh.

Steve: Gizmodo said, "WPA Wi-Fi Security Gets Cracked: Your Network Is No Longer Secure." And, I mean, the headlines have been blaring because, I mean, this is, well, it would be big news if it were true. Something did happen. It was significant, but - and also incredibly limited and incredibly clever. So this ends up being a major propellerhead episode. What's so cool is that we can explain what happened in a way that people will understand. And there's remediation things that people can do. Some things that people might do wouldn't work. So anyway, we're going to explain exactly what it was that was figured out, who did it, what they did, and what it means.

Leo: All right. The details of the exploit and, most importantly, what you do about it to avoid it. And I've already done that, by the way.

Steve: And the correct story. I mean...

Leo: Yeah, that's the most important.

Steve: The absolutely this is what it is. Anyone who listens to this podcast will come away really getting it, I mean, what exactly this thing is. And it is complicated. So we're going to have a good episode.

Leo: All right. We geek out. So, Steve, before we get to this TKIP hack, is there anything you want to cover from last week, or security news?

Steve: Yup, got a bunch of little goodies. We actually have some errata bin things. One of the notes I read as I was reading through last week's Q&A - actually I guess I saw it afterwards or I would have mentioned it then - was somebody felt that I was evidencing a bias against Russia and China.

Leo: That's a reasonable point. And...

Steve: Yeah.

Leo: Yeah, I think that's a reasonable point.

Steve: Well, I would say it's reasonable if it weren't - it would be reasonable to believe I was biased if perhaps I wasn't clear enough that in fact I'm repeating and restating geographical fact about where these attacks originate from. I mean, unfortunately, for whatever reason, and I don't have any political bias, these attacks actually come from China and Russia.

Leo: Well, but I think we should make the point, I think we've made it before, that doesn't mean that's their origination point. It could very easily be that somebody's hacking from Dover, Delaware, but going through a Chinese server.

Steve: Yes. In fact, studies have shown that, for whatever reason, again, lots of Chinese machines are compromised by zombies. And so when, for example, denial of service attacks come, they come in from machines at Russia, not because - or, I'm sorry, from China, not because Chinese people are launching them, but because Chinese people have their machines compromised. So...

Leo: Possibly because they have a lot of - they apparently have been using a lot of pirated versions of Microsoft Windows. They're probably not getting updates. And of course they're getting hacked as a result.

Steve: Right.

Leo: But it happens everywhere. But I guess the real point is that you can never determine who's doing the hacking based on where it seems to be originating from. In fact, it's highly unlikely that it's originating from the last point of departure. You'd be a bad hacker to say I don't care if you know where I am.

Steve: Right. Although also, I mean, it is known that there is organized crime in Russia that is responsible for a lot of the cybercrime that we see. I've had several conversations with FBI friends who have, I mean, who've backtracked this to specific organizations, to specific locations physically in Russia. So when I do say something is originating from this or that country, it's normally that the evidence and the facts - I'm not making that up; I'm not picking on them for no reason. It's that typically that's really where a specific event that I'm describing came from.

Leo: Well, and to follow our theme of today, the mainstream media doesn't make that - very frequently doesn't make that observation. When you saw the stories this week that the Obama, McCain, and White House - Obama and McCain campaigns and the White House had been hacked from China, the implication was it was Chinese hackers. But that does not mean that at all, does it.

Steve: No, it could easily be that somebody was relaying the attack through a machine located there. And you would, because for the kind of penetration that was found by the

FBI, they end up with logs and IP addresses. And you know somebody is getting their door knocked on in China. So hopefully it's an innocent grandmother who has her machine and didn't understand what was going on, and on it they will find some relay technology that has left no track or trace for where the actual connection came from. So somebody was using it to cover their tracks.

Leo: When I do a whois - I get attacked all the time, I imagine any server does - and I see IP addresses in my log, usually it's people trying to brute force SSH or other services. Whether or not I'm running them, by the way.

Steve: Or it's, as we've discussed, it's Internet background radiation. It's just junk that we know will never die, probably.

Leo: Banging on my door. Well, I see, you know, they type SSH, you know, root at TWiT.tv, and then they try a password. It doesn't work, they get three chances, they come back. Usually it is a Chinese IP address coming from a university. That's what's interesting when I do the whois. It almost always the address is owned by a university in China. That could be two things. Could be a college kid, or - which is, I think, perfectly likely - or it could be that that university, like many, have UNIX servers or other servers that are easily compromised, they're just on all the time, and that's frankly where a lot of attacks come from. Doesn't mean that's where the attacker lives. Anyway, that's my point.

Steve: The other thing that - the other observation which people in our newsgroup made, and I even saw people submitting their observation, the similar observation to GRC.com/feedback, is I made the comment last week when the guy was talking about how he runs netstat, the netstat command, and he had to resort to, like, hitting PrtScr really quickly because...

Leo: Yes. Doh.

Steve: You know, it didn't even occur to me that he may have, and this has been suggested, been typing "netstat" not at an open DOS prompt window, but under the Start menu, using the Run option, to get the little command line, typing it there. Well, when you type netstat - "netstat," for example, space "an," in the little Run line under the Start dialogue off the Start button, it will launch the command window, run the command, and shut the window down afterwards. And it's like, oh, I'll bet that's what he was doing. Just didn't even occur to me. So for what it's worth...

Leo: It occurred to all of our listeners, I might add. I think I got a lot of email. And every one I went, oh, of course.

Steve: Exactly. So, for example, if you wanted to launch the command in a static way, in the same way, from the Start menu, you could start by saying cmd space /k. "K" keeps the window up after the command has been executed. Then say "space netstat space hyphen an," and it'll launch the window, run the command, and then leave you there.

Leo: Oh, that's a good way.

Steve: In fact, I think it leaves you with a "Press any key to close the window." So you can then scroll around, look at it, see what you want to do, then hit Spacebar or Enter, and it'll close the window, and you're back where you were before.

Leo: That's a really good idea. I never thought of that. I always just type "cmd return" to open a window, and then run the command in there.

Steve: Right. So in security news, this is a podcast occurring on the Thursday after the second Tuesday of the month. And we know pretty much reliably every month what that means. That means that Microsoft has released some security updates. This is no different than most months. Not a huge number. I think once we had 11. This time we just have two. They are both remote code execution flaws. Microsoft had a whole bunch of problems with their XML parsing, XML being sort of an interesting standard for flexibly and in textually describing hierarchical relationships among data. And there were some code execution problems there, one critical, one important. So standard routine is, as always, just make sure that your machine is up to date. You'll want to get to that sooner or later.

It's also worth mentioning that the flaw we talked about a couple weeks ago now, I think it was two weeks ago we talked about a problem with Adobe PDF file parsing. It is now being actively exploited. So you want to upgrade to v9 if you can. I don't know if Adobe has an update for - I don't think they updated beyond 8.1.2, which was where the flaw, anything there or prior. So you do want to move to v9 of your PDF reader, which you can easily do just by going to Adobe.com, as I mentioned before, and download an update. But there are websites - apparently ads, web ads in bad sites are carrying a reference to this PDF. Your browser will load it and get itself taken over, and trojans are being installed through this vehicle. So it's something you do want to take care of. And...

Leo: Adobe updated Flash, too, for a similar reason, didn't they.

Steve: Yes, yes. And finally, I know you've mentioned this. I've watched you on TWiT Live, Leo. But it's worth mentioning that Google has updated Android to fix a very embarrassing problem.

Leo: No kidding.

Steve: With the first release of their phone.

Leo: It happened on my phone. I couldn't believe it when I read it. And I immediately typed "reboot" into my phone, just at the, you know, not at the command line, but just at the desktop, you know, when you first turn on the phone. The phone reboots.

Steve: It turns out that they left a debugging switch set in their final release build of Android, such that anything you entered through the keyboard was going also to the root shell of the OS, with root privileges. So, I mean, so no matter where you were, you could be texting somebody or entering data into a file, I mean, anything, it was all being echoed into the root shell. So you had root level command privileges by default. And as you said, Leo, you type "reboot" anywhere and hit Enter, and the phone shuts down. Whoopsie.

Leo: It's the strangest bug I've ever seen. And I was stunned that it worked. And I got - I have to say that immediately, the day that article came out describing it, a patch was pushed. And I think they've pushed it now to everybody with a G1.

Steve: Yeah, yeah.

Leo: Terrible, though.

Steve: Well, embarrassing more than anything. If it's your own phone, it's not any kind of remote exploit. But still, you know...

Leo: Well, I imagine that probably little hacks, you know, people said hey, here's a special cool thing you could type into your - the other thing that was a little worrying is it appeared that it was logging everything that was being typed. So in theory there's a file there on that phone that had some passwords and things like that.

Steve: Into the history buffer, right, that's a very good point. Even passwords and things, yes. And a listener of ours, Bob Morris, sent a nice little email saying "SpinRite Success Story." This is a quickie, but a nice example of what SpinRite was able to do for one of our listeners. He says, "My aging P4 went south a few days back. Chkdsk reported errors on D:, circular redundancy in a folder, and some other such errors. So I ran chkdsk D: with the /repair option. Then the drive was gone completely. It asked me if I wanted to reformat. Yikes. So I ran SpinRite on all drives. SpinRite said S.M.A.R.T. was reporting the drives' imminent peril. Did I want to tax them any further by continuing?" He said, "I said yes because there were still a few things on the hard disk that I hadn't backed up. SpinRite ran for 10 hours. Upon rebooting, while S.M.A.R.T. still reported problems, D: was back, and I immediately backed up all my remaining files. Thanks again, and I listen to your excellent podcast frequently." Signed, Bob Morris.

Leo: Isn't that nice.

Steve: So nice little happy SpinRite success story.

Leo: Thank you. Bob. Might take a break here, and then we're going to talk a little bit about what this exploit, this WPA exploit is.

Steve: Oh, we're not going to talk a "little bit" about it.

Leo: Explain what it means.

Steve: Buckle your seatbelts, folks.

Leo: And, now, the mainstream media got it wrong. It has to do with this temporal key thing.

Steve: Integrity protocol, yes, TKIP, a mistake in basically, probably maybe by default, the WiFi encryption that everyone is using because until now it was believed to be completely safe, and it was less taxing on systems and more widely compatible because it's an evolution upwards from the, as we now know, really badly broken WEP encryption that nobody should be using. The problem is, it's not quite as bulletproof as it was believed. It's not broken, by any means. It's not - this is not some horrible end-of-the-world problem. But we're going to explain exactly what it is.

Leo: You can imagine my consternation after recommending and telling for the last two years everybody, all you need on WiFi is WPA, you're safe, to read that headline, "WPA Cracked." So we'll tell you what it really means in just a second.

[Commercial break]

Leo: So, Steverino, it's time to get down to WPA brass tacks.

Steve: Yeah. Okay. So I would imagine that our listeners have seen the stories all over the place. Most of the electronic online media carried the story. This was big news. The distressing thing, the reason we're talking about it, is that I read everything that I could find. Nothing got it right. Nothing...

Leo: Even PC Magazine.

Steve: Yeah. PC Magazine was as bad as any, actually. I mean, really, I don't know why everyone went overboard about this. Maybe it's they didn't understand what it meant. And I can forgive them for that because this is a very complicated - it's complicated to execute. What you end up being able to do is very limited, but potentially means something. Anyway, so...

Leo: Now, has he revealed the technique? Because I...

Steve: Oh, yeah. In detail. We know everything about it.

Leo: Okay. Because the initial stories - and maybe that you can't fault the initial

stories because he hadn't explained how it worked, he had just merely said there's a hack, and I'm going to tell everybody in a week.

Steve: No.

Leo: No?

Steve: What I saw from day one was it being clear - well, okay.

Leo: He was going to present it at some security meeting, I think.

Steve: Well, actually that's happening right now. Given that this podcast is being listened to on Thursday, November 13, it is - Erik Tews is the lead on this. His friend Martin Beck, who - they're both students in Germany. Erik is a Ph.D. candidate at the Technical University of Darmstadt. And Martin Beck is a student at the Technical University of Dresden. And Erik is the guy who brought us the 60-second WEP crack.

Leo: Oh, so he's an expert.

Steve: So he is - oh, yeah. In fact, we had a podcast about this. I think we called it "More Badly Broken WEP." It was already damaged. But Erik demonstrated how it was possible to obtain the key, the WEP key that is in use on a WEP-based WiFi network, full WEP encryption, how you could obtain the key with as little as 25 and as much as maybe 40,000 packets, which on a typical network would take about a minute. Now, prior attacks, there was an earlier attack that took about 700,000 packets. So, yes, you could still get the key. But, I mean, these guys really, really know this stuff. So that was one reason why they were taken - why what they said was taken very seriously. Now, it's a - I would call this an extremely, well, I'm going to explain exactly what it is and how it works. But it's not WPA is broken.

One of the other problems is there are a bunch of acronyms we're swimming around in here. For example, I've read online that WPA is not safe, but WPA2 is safe. Okay, that's not true.

Leo: Oh, it's not.

Steve: No. I mean, because WPA and WPA2 are not cryptographic protocols. They are certifications offered by the Wi-Fi Alliance of certain levels of operation of WiFi hardware. So WPA2 is not something different from WPA. It's a different level of certification. But it's not - but WPA2 can be just as insecure as WPA.

Leo: See, I thought WPA2 used - automatically used AES. So it could use TKIP.

Steve: No. No. And there again, AES is a cipher called "Rijndael." TKIP is a protocol. CCMP is the protocol which uses AES. I mean, so my point is...

Leo: I see. So I, too, have completely misunderstood this.

Steve: Right. So, I mean, there's so many acronyms here. So we're going to go through it very carefully. And again, I promise our listeners, when they stagger away from this podcast, they'll know exactly what is going on and what has happened and be able to tell all their friends, wait a minute, here's what you have to do, and this is what this means. And except for, as far as I know, Security Now!, everything else that's been written so far just doesn't, you know, people are doing the best job they can. But this is complicated. And acronyms and the usage of these technical terms correctly is really important.

So let's turn the clock back first and remember how WEP works and what's underlying it because that's the source of the breach that these guys have been able to create. The way original WiFi encryption, WEP, worked - remember that stands for Wired Equivalent Privacy. The idea was that the original designer said, well, we're not saying that this is the end of all possible problems. But we're going to give you privacy that's the equivalent of what you would get with a wire. It's wired equivalent. And it turns out they were wrong about that.

The way it works is there's a very simple pseudorandom byte generator. And we've talked about how XORing encryption works. If you took a message composed of a stream of bytes, and you were to XOR that stream, we'll call that the plaintext, the normal, unencrypted text. You XOR that with random noise. What happens is, the way the XOR operation works is, bit by bit, if the bit is on in the noise, it inverts the bit in the data. And if it's off in the noise, it does not. So what that means is the noise selectively inverts the data bits to produce the cryptographic result. And odd as that is, I mean, as simple as that is, if you really have noise, that is, random noise, that means you are randomly inverting the bits in the data. And it turns out that there's nothing that can decrypt that. Nothing. I mean, as simple as that is, nothing can decrypt it. So the only thing that can decrypt it is if you take the same noise again, exactly the same noise, and do the same thing to it. You XOR it, which is that operation. It's what the operation is called, an exclusive or.

And if you think about it, if you have the same noise, and you invert the same bits again, then the bits that you inverted have been inverted twice. They were inverted to encrypt it and then inverted to decrypt it. And the bits that weren't inverted just kind of went right through. So if you invert a bit twice, you get the same bit out that you started with. If you have a zero, you invert it to a one, and then again back to a zero. If you start with a one, and you invert it to a zero, and then again back to a one, it comes out the same way. So you can see that this XOR operation is trivial. And two of them essentially remove themselves. Two is the same as none because it's that simple, conditional bit inversion. So the beauty of that is it is incredibly simple to do this in simple hardware. And the original WiFi specification really wanted to be able to implement WiFi with minimal hardware, or minimal firmware or software. So they came up with a simple source of random stuff.

Now, we switch from the notion of actual noise, actual random stuff, to pseudorandom data. The pseudorandom data means you've got some algorithm of some sort which is complicated enough that it emits data bytes that appear random. That is, analysis of them does not give you an obvious pattern, so that somebody can look at data bytes

coming out of this algorithm, and to them they look random. Now, the algorithm in use is something called "RC4," which was developed by RSA a long time ago and kept as a proprietary algorithm. Technically, they never formally released it. It was always a trade secret. But it leaked out in the world, and everyone knows what it is. It's a really good algorithm, but it's got some problems, which is it involves a 256-byte array, which is sort of mixed up and scrambled based on the key you give it. It turns out that there are weak keys that don't do a good job of starting off with this array being scrambled. And also that, since this array is scrambled as it works, some of the initial data that it produces, the so-called "pseudorandom data," isn't as random in the beginning as it ends up being later on.

So all of these things created some weaknesses in the original implementation of WEP. But the idea is pretty simple. You have a key. And you use the key to produce a stream of pseudorandom data using this RC4 algorithm. You XOR your so-called "plaintext," the normal packet data, with the output from the pseudorandom generator. And it's going to flip the bits. It's going to randomly, pseudorandomly, invert the bits in the source data to create something that's encrypted. And again, as long as no one knows what the output from the pseudorandom number generator was, what you get out is also pseudorandom. There's nothing that you can do to figure out what the original data was except reinvert the bits with another copy of that same pseudorandom data.

So that's how original encryption worked, is both endpoints would have a so-called "preshared key." They would both know what the key was, so they would know how to generate the pseudorandom data. The one sending would generate a stream, a so-called "key stream," which is the stream that's generated by the key. It would take the key stream, use the XOR operation to invert the bits in the plaintext, the unencrypted packet, stick it out in the air. The other end would receive it from the air; and, if it had the same preshared key, it was able, by using that, to regenerate the same pseudorandom sequence, the same key stream, XOR what came out of the air, the encrypted data, with the key stream, which is the process of reinverting the bits that the transmitter had initially inverted. And in doing so, as you can imagine, as you can see, it gets back the original, unencrypted data. So that's how WEP worked.

Now, there were some complications that we don't need to go into. We have described them in detail in earlier podcasts, if anyone's interested. For example...

Leo: I encourage people to listen to that podcast because that alone is enough for a whole show.

Steve: Yeah, and it was. For example, one of the weaknesses - there are a number of weaknesses with something as simple as XORing because, if you knew some of the plaintext data, that is, if you knew some of the data that had been encrypted, for example, if you knew the IP address, the source or destination IP address that would be in the packet, well, if you XOR the encrypted data with the unencrypted data, what that gives you is the key, that is, the key stream. That gives you the pseudorandom data that was used to invert the bits to create the encrypted data. You can work it out on a napkin, if you're curious. It's sort of cool. And it means that there's, like, there's three things. There's the unencrypted data, the key stream, and the encrypted data. And XORing any two of those gives you the third. So if you know the cipher text and you know the plaintext, you can get the key stream. And it turns out that that's something we're going to come back to because that's part of what this hack involves. So the point is that you never want to encrypt different packets with the same key stream because it's very possible then to find correlations between the packets and start the process of

decrypting.

So one of the things - there are many other additional complexities to WEP that, again, I don't - I'm not going to go into here because they're not really germane to this. But, for example, there's a 24-bit counter on the front of the packet which is used as part of the key in order to prevent successive packets from ever having the same key stream. And so when the receiver gets it, it looks at that and is able to figure out exactly what the key was that was used so that you're not actually reusing the same key, the same key on the same packets, because you don't want to do that. So in order to verify that there was no transmission error, at the end of this packet four bytes are added. And it's just a standard CRC32, a 32-bit, that is to say, four-byte CRC, a Cyclic Redundancy Check, which is a well-known algorithm that was added to catch any transmission errors, literally bursts of static in the air that would cause the receiver not to receive what the transmitter sent. So after the packet is decrypted, then the packet is scanned, and the proper CRC is computed. And that's called the ICV, the Integrity Check Value. And that was part of the original WEP also. And it was often implemented in hardware because a CRC is easy to do in hardware, as was this RC2 pseudorandom number generator.

Okay. So as a consequence of the relative simplicity of the system, all kinds of problems were found. One of the problems that was found, and we talked about it, there are some clever ways that it's possible, essentially, to determine what the original key is, that preshared key. And if you do that, then you know what the same - you have the same key that everybody on the network has, and you're able to then receive any encrypted traffic and decrypt it, and generate your own spoofed or false traffic, encrypt it, and send it out into the air, and everyone will believe it because, when they receive the packets and decrypt using their key, the packets are going to be valid.

There's a different kind of attack which a clever hacker whose name no one knows came up with. He uses the handle KoreK, or maybe it's Kore K. It's K-o-r-e, and then capital K again. He came up with a really interesting attack called "chopchop." It's named that because he realized you could chop a byte off the end - and we're still talking about WEP now. We're talking about the original Wired Equivalent Privacy because, as we're going to see, some of these problems unfortunately ended up surviving as we moved into the world of WPA and more complex protocols that were designed to prevent these simple attacks. So this KoreK guy, he realized that, if you chopped the last byte off the packet, it would almost certainly now be invalid, that is, the ICV, the Integrity Check Value, was now no longer going to work correctly because you had chopped off the end of it. But he worked out exactly what the relationship was between the first three bytes of the Integrity Check Value and the last one that you had chopped off; and the fact that, if you sent that back out onto the network, now the system would think that the last byte of data was the first byte of that ICV, the Integrity Check Value. Remember, because the ICV is always the last four bytes of the packet.

So this guy worked out a way of getting the access point to tell him what that last byte was that he'd chopped off. Because the access point will complain, will send back a message saying, wait a minute, you've got a checksum error. And so the idea was that, since a byte can have any 256 values, from 0 to 255, that you could simply guess, make guesses about the value of that last byte. And in an average of 128, that is, in an average of half the guesses, you would end up having the access point confirm that you now knew what the byte was because you had corrected the checksum.

Leo: So you're allowed to keep trying until you get the checksum right?

Steve: In WEP you are.

Leo: That's a flaw, obviously.

Steve: That's a big flaw in WEP. And they fixed it. But not quite. And that's part of what is so clever about what Erik and Martin figured out in WEP - I'm sorry, in WPA. In TKIP. So in WEP you could flood the access point with these invalid packets, and it would just dutifully tell you if you had guessed right or not. And so very quickly you had figured out what that byte was. Then you chopped the next one off, and you figured out what that one was. Then you chopped the next one off, and you figured out what that one was. Now, remember that that checksum testing is done on the unencrypted data. That is, the way this works is the packet is received. Decryption is applied. Then you have the so-called plaintext, the decrypted text. And it is processed to see if the checksum matches. So the checksum isn't...

Leo: [Indiscernible] another flaw. Couldn't they do the checksum on the encrypted text?

Steve: Well, perhaps. Who knows, I mean, I haven't thought that through, what that would mean. Although you would probably...

Leo: It would eliminate this hack.

Steve: Well, it would change the hack. See, the problem is that this integrity check value is too simple. It's just four bytes, and everyone knows what the algorithm is. It's not even keyed. There's no unknown data for it. It's just a standard algorithm to check - and, see, that's the problem, is it was meant to check for transmission errors, not meant to check for the data being spoofed, not meant to check to see if the data were changed. It wasn't a so-called - I'm blanking on the name now. It wasn't meant to authenticate the packet, merely to check for changes. So anyone could make changes, and then change the ICV to make the CRC again valid. That's trivial to do. But the problem is, it's encrypted.

But here's the cool thing, is that by guessing the byte which is unknown but encrypted, once the access point tells you you've got it right, well, you know what the byte originally was because, remember, that's the byte you chopped off. So if you XOR your guess, which is the decrypted guess, with the byte that you had, what you get is the key stream byte, that is, this is a way, by using chopchop, you are not only figuring out what the plaintext decrypted value is, but by comparing that to the encrypted value you get out what the key stream is.

So what this means is that this KoreK guy figured out a way of taking any packet that is received under WEP and successively chopping off the last bytes, walking this packet down in size, and he's going to end up getting the - decrypting the packet, determining what the plaintext was. And here's the cool thing. He gets the key stream. That is, he gets the exact pseudorandom sequence which was used to originally encrypt the packet. What that means is he can make up his own packet, encrypt it himself with the same key stream, and inject it into the network. He doesn't know what the actual preshared key is. He was just able to figure out one instance of the pseudorandom data that had been

used for that one particular packet to encrypt it. But that's all he needs because he's able to take that and make his own valid packets.

Now, what that means is there's no replay protection in WEP. There's nothing to prevent you from basically making an altered packet where the sum of the packet is the same, for example, sum of the header information may be the same. But you're able to change the data, create a correct CRC32 for it, that ICV, the Integrity Check Value, and stick it back out onto the network. And everybody's happy. They receive it. They decrypt it using their preshared key, which they know. You still don't know what that is. You don't need to because you've got a sample of the key stream which was valid for that particular packet number. Remember there's that 24 bits at the front that says this is the particular key stream for the following packet. You use that, and you're able to just spoof packets and synthesize them.

So, okay. So all kinds of problems are surfacing with WEP. And the IEEE decides, okay, we've got to come up with a solution for this. So they start working on a spec called 802.11i, which is going to be the security portion of the 802.11 overall specification for wireless security. But when you look at the number of people on this 802.11 committee, I mean, this thing goes - it goes on for pages. And it's the reason that this thing took forever to do. It's a huge committee. Nobody could agree on anything. And so the whole industry is sitting around waiting for the 802.11i specification, and finally gave up waiting. They said, you know, we can't wait any longer. We need a solution for this. So we're going to come out with something called WPA, which is going to be a certification for some of what you guys have agreed on so far, basically based on a preliminary incomplete version of the 802.11i specification. You've figured out this TKIP. You're talking about maybe using AES for stronger encryption, but you haven't figured out exactly how to want to do that yet. Well, TKIP is a whole lot better than WEP, which is so badly broken now, so we've got to get on with this. We're going to just go with TKIP. So what happened was a bunch of manufacturers didn't wait for the 802.11i specification to get finished. So they came out with the TKIP portion of a next-generation wireless.

Leo: Interesting.

Steve: And they said, well, you know, it looks like AES, which uses the really well-regarded Rijndael cipher, that's going to be part of it. And here's maybe how it's going to work. So we're going to toss that in. So...

Leo: I can see the problem already.

Steve: Oh. Well, it turns out that things did change in the AES side of the 802.11i spec, between the time the manufacturers launched out of the starting gate prematurely and the time it was finalized. So that initial hardware that said it was WPA certified, and oh by the way we've added AES because it's that good, that turns out it won't work necessarily. There's several things different about that than the final specification, which is what WPA2 certifies.

So to clarify that a little bit, or say it differently, WPA is not encryption. It's not a protocol. It's not a cipher. It's nothing but a certification from the Wi-Fi Alliance. And WPA certification means that your system can run TKIP properly, in full conformance to the 802.11i IEEE formal final specification. Because that didn't change from the time all the manufacturers went out of the starting gate prematurely. So WPA just says it will

interoperate, equipment that is WPA certified will interoperate with TKIP protocol security.

Now, again, many of the hardware devices, access points, for example, at the time also threw in AES. And you may have seen, some of them say AES-64, AES-128, AES-256. Well, there is no AES-256 in the final spec. So any hardware that had that is completely non-interoperable with the WPA2 certification that was finally arrived at. So that's an example of where manufacturers sort of got a little bit ahead of themselves. But the TKIP portion was solid and has remained so in terms of its specification.

Okay. So in terms of terminology, WPA says you've got TKIP. What is TKIP? That's an acronym for Temporal Key Integrity Protocol. The guys at the IEEE, this massive committee with an unbelievable number of people, they said, okay, we know we need to fix all these problems with WEP. Let's do so. So they did a number of things. They added a replay capability, that is, an anti-replay awareness to prevent any kind of replay attack, so that you couldn't take a packet and either send the same one in later and just replay an identical packet - it turns out there are even attacks where you don't have to know anything about the packet, but just sending the same packet later can cause problems. Literally, you decrypt nothing, you figure out nothing, you just inject it later, and it messes things up. So they said, okay, we want to prevent that from happening. We also need to fix this really weak CRC that's hanging on the end of these packets because that's dumb. It's only useful for checking for mistakes. It's not useful at all for checking for malicious packet contents modification.

So they came up with a new double-size, this thing's eight bytes, thing called an MIC, which stands for Message Integrity Code. And actually it's known as Michael, just M-i-c-h-a-e-l. So it's the Message Integrity Code. Now, they put the Message Integrity Code first, and then the ICV, the Integrity Check Value, at the end, again because their goal was to make TKIP upward compatible with existing hardware. This actually is the flaw. The fundamental flaw in all this is that they tried to wrap improvements around a really fundamentally insecure approach for WiFi, which was WEP. But they did it with the best of intentions. They gave us all years of pretty, I mean, much better security than WEP for all of us who have routers that are using WPA and TKIP protocol, the TKIP security protocol on WPA-certified equipment. So they made it much better.

Okay. So the problem is they're still using TKIP. The Temporal Key Integrity Protocol still uses RC4, that is, it still uses that pseudorandom sequence generator approach. And it still uses this XORing of the pseudorandom data with the plaintext approach. This was, again, done deliberately to create something where you could just upgrade the firmware. But, for example, in much of the WiFi hardware RC4, that pseudorandom sequence generator, was built into the hardware. And that CRC32 was built into the hardware. No matter what you gave the hardware, it would tack on a CRC32, this ICV, the Integrity Check Value, automatically. You couldn't make it not do that. So they said, okay, that means we need to put a better integrity code at the end of the packet, which the hardware will then stamp its CRC32 onto the end of. And so they designed this very cleverly to be able to be retrofit into existing systems that only understood WEP. And they succeeded. And it's been good for a number of years.

Meanwhile, the 802.11i committee kept cranking away, and they settled on all the details of using an entirely next-generation approach, AES, the so-called "Rijndael cipher," which is extremely robust and good. And they didn't have to worry about the past at all. They dealt with the past using TKIP. And they said, so we're going to have two different security suites in WPA2. You did not have to have any AES in order to get WPA, the first WPA certification. That is, the Wi-Fi Alliance said we don't really know how that AES thing is going to work out because the 802.11 committee is not done yet. So WPA only had to

have TKIP. WPA2 has both. And so it's not the case, for example, that WPA2 no longer has TKIP. It has both. It's got the final version of the AES approach. And I'm saying AES. AES is the same as RC4 over in TKIP. The protocol that uses AES is something called CCMP, that's an acronym, because the way the AES cipher is used is in a way called counter mode cipher block chaining. So CCMP is the protocol that uses AES in the same way that TKIP is the protocol that uses the RC4 cipher. So very likely in today's access points, anything that you've purchased recently that is WPA2 certified, that's the Wi-Fi Alliance saying we've tested this equipment using both TKIP and AES, or I should say CCMP, although unfortunately the user interfaces of these all say typically TKIP or AES, even though one is a protocol and one is a cipher. To be really accurate they should say RC4 or AES, or they should say TKIP or CCMP. But they don't. Now you understand exactly what these acronyms mean.

So we've got ongoing research, then, by these guys, Erik and Martin and KoreK, into various ways of screwing around with TKIP. The question is, we know it was based on old technology, deliberately keeping some of the requirements of the old hardware so that we could fix WEP without obsoleting all of our investment in hardware, so that just driver software or firmware could be changed. But if the hardware was going to stamp every packet with an ICV, an Integrity Check Value, we had to allow it to still do that. And if the hardware insisted on generating, you know, using RC4 to generate pseudorandom data and just XORing it with the data, we have to somehow make that work. So they were able to change sort of the interior of the packet and leave the exterior envelope the same.

One of the things they did was they added, as I mentioned, this double-size, this eight-byte MIC, this MIC, the Message Integrity Code, to the end. Now, this was a much more powerful solution than CRC. It uses a key. So it's a keyed authentication chunk. And unless you know what the key is, you are unable to synthesize the proper eight bytes to authenticate the payload of the packet that precedes it. And so they were able to sort of change the interior of the packet in order to keep everything else the same.

Well, these clever hackers figured out how to use a chopchop-like approach on TKIP. And here's the way that works. So now we have a WPA or WPA2, remember, those are just certification levels. Those are, as I say, nothing about which security protocol you're using. So either WPA or WPA2, both of those will have TKIP. And many people have been using that because of its backward compatibility and because it's been felt to be good enough. There weren't any known problems with it until now.

So we've got that scenario. And so we capture one small packet out of the air. The length of the packet turns out to be important because it's necessary to know what most of the packet is. It turns out there are many limitations in the nature of this reverse engineering hack that these guys came up with. But you can do some damage to a network even with small packets. For example, ARP, the Address Resolution Protocol, that's the protocol which matches up the physical adapter addresses, the so-called MAC address, to the logical Internet Protocol address, the IP address. So ARP is the glue where you're able to send an ARP packet out onto an Ethernet network and say who has this IP. And all Ethernet adapters listen for these ARP broadcasts, and they check to see if the question is for them. Oh, I have that IP. In which case they send an ARP reply back to the MAC address that issued that ARP broadcast.

So these are very small packets with very well understood format. And on a given network, not that many bytes are unknown in such a small packet. So this attack on TKIP begins by somebody grabbing just one of these small packets off the air. And you can pretty much know, you know exactly how long it's going to be because you're going to have the ARP data with well-known ARP headers and ARP contents, followed by the

eight-byte MIC and the four-byte ICV, which is the format of these packets in the air when they've been encrypted.

Well, it turns out that, if you do the - you start doing the chopchop guessing. You chop the last byte off the packet, and you send it back out into the air, back, for example, at the access point. If the checksum that you guess - remember they still have an ICV on the end. If the checksum is wrong, a TKIP - a newer, modern, strengthened, better protocol system - if the checksum is wrong, it ignores it. It just says, bad checksum, I'm dropping it.

Leo: And you don't get another chance.

Steve: No. It simply drops it because it figures, okay, that was a transmission error. It figures it's a transmission error, so it doesn't punish you for that. So with an average of 128 guesses, just like before under WEP, but now we're under TKIP, using the same kind of approach, when you get it right, when you do end up creating a shorter packet with the CRC, that is the ICV at the end that matches, now the problem is the MIC, the Message Integrity Code, will be wrong. And now that, when that's in violation, if you get a checksum that's correct, but the MIC, the Message Integrity Code, is wrong, now you've pissed off the access point or the client you're sending this to. Anybody who's receiving it is like, whoa, wait a minute, this is a valid packet, but the MIC is wrong. Something's fishy somewhere.

Well, they didn't want to just shut down the whole network. So they said, okay, here's what we'll do. As long as we don't get two MIC failures within a 60-second window, as long as they don't occur more often than once per minute, we'll decide that's okay. Whoops. Because look what happens. You can guess as much as you want and be wrong. But as soon as you guess correctly, you have to wait a minute. But that's not so bad because you just guessed correctly. In knowing that you have to wait a minute - because what happens is a message is sent out that says "MIC failure," so the whole network knows there was one, to sort of like put everybody on notice. But you've just been put on notice that you guessed correctly. So you've got one byte. So you wait a minute, and you start guessing the second from the last byte until you get it. Now, that allows you to march the packet down in size 12 bytes. And that'll take a little over 12 minutes. When you've done that, you've just determined the plaintext for the MIC and for the ICV. Remember, those were the last 12 bytes on a TKIP-encrypted packet.

Okay, so now you know what the plaintext for the MIC is, the Message Integrity Code, and what the plaintext for the ICV is. Now you can guess the other few things that may be unknown, like the IP addresses of the sender and receiver, by plugging them in and checking to see whether the ICV matches. You didn't know what they were. They're up at the front of the packet. But now that you know what the ICV is, you can perform - and the ICV is a simple CRC32 - you can plug in, quickly plug in guesses until you get a match. That allows you to determine what the IP, the source and destination IP was.

Okay. Now you know all of the packet up to the MIC, to where the original, full-length Message Integrity Code was. Well, turns out that the Message Integrity Code was never designed not to be reversible. That is, it's not like a hash, where you cannot reverse it. It's an algorithm that is as easy to run backwards as it is to run forwards. Knowing what...

Leo: Like an XOR.

Steve: Sort of. It's fancier than that. But it is reversible. So now, knowing all the data ahead of it, which would be the input to its algorithm, and knowing the result, you can reverse engineer the key, the so-called "MIC key." So now you have the MIC key which was used - which is unique for, not per packet, it turns out, but it's unique for a keying session. So that allows you - you've got the MIC key. You know all of the plaintext of the packet. And as a consequence of knowing all the plaintext, remember that, since we also captured the original cipher text, the encrypted text, you just XOR those, too. Now you've got the key stream. That is, you've got a sample of the TKIP key stream that you were never supposed to be able to get. And because you've got the key for the MIC, the Message Integrity Code at the end, you can now make up any kind of packet you want, ahead of the MIC, and recreate a proper MIC, which will then pass muster.

Now the final problem, and this is the last bit of just stunning genius from these guys...

Leo: Yeah, because you still don't know how to decrypt all the traffic.

Steve: Oh, you never do. Yes, this does not give you that. What you've got is the ability to make up, to modify...

Leo: To inject stuff.

Steve: Yes. Well, and it turns out not much, because they did add in TKIP, they added a block against replay. There's replay attack prevention. There is a counter that they added to the packet. And every time a packet is received, that counter is incremented. And so there's going to be continuing packet traffic, and that counter is going to keep incrementing. So if here you come along and say, hey, here's a packet, and you try to reinject this into the network, it's going to say, sorry, we've already been there. We've done that. That's an old packet. Don't know where you came up with it, but we're not interested.

Okay. Get this. These guys realized that in a system that has Quality of Service support - 802.11e is the Quality of Service support - that that represented a breach in replay attack prevention. Okay. What Quality of Service is, is it says, okay, on Ethernet networks there may be times when some traffic needs a higher quality of service. For example, VoIP, Voice over IP, like you and I are using now with Skype, Leo, it's not that it needs, like, to dominate all the bandwidth. It's that it's delay, it's delay sensitive. You know, Voice over IP, we need to know that those packets are streaming out, and they're not going to be buffered up in some queue, waiting for bandwidth. They're going to have priority. So it's a priority system.

But in order to do that, you do have to have queues. You have to have buffers where other packets can be held while the express train packet, the VoIP, the higher priority packet, is able to pass by and get through. So in most modern access points which support Quality of Service as one of their bullet points, oh, look, you want to buy ours because we've got QoS on ours, what they've got is up to and typically eight buffers. Most traffic just runs on channel zero, and channels one through seven are not used. Well, that means their replay counters are not incrementing. And when you send one of

your made-up new packets to a different QoS channel, it will be accepted rather than rejected because...

Leo: Wow. You get eight chances.

Steve: Yes. And so, exactly, you could do the decryption once, and you can't use the channel that it came from. But you can use the seven others. So you can then - you decrypt the packet once, and that takes - it's going to take 12 minutes for you to get those last 12 bytes, one at a time, because you remember you're punished by having to wait a minute. And if you don't wait a minute, that sets off alarms in the whole network that causes the access point to shut down for 60 seconds and then rekey everybody. So you've lost all your work unless you make sure that you wait at least 60 seconds between succeeding with one of your guesses because the succeeding with the guess means that the message integrity value which is inside the packet will fail. And that sets off the alarm. But that's okay because it just confirmed that you guessed the last byte correctly because you got the checksum correct.

Leo: Right.

Steve: So what this means is, with TKIP on a network, that after 12 minutes it is possible to take a small packet - and the reason it has to be small is they have to know everything about the packet. That is, the MAC addresses are not encrypted. They exist at the front of the packet before the encryption begins because you have to have the MAC address in order for it to come or go. A packet like an ARP packet has very little data that's unknown. A long packet, you don't know what's in there. And so there's no way for you to, well, I guess you could continue marching down the packet one by one, pissing off - for every single byte having to wait a minute. But for a typical full-sized packet, which is 1,500 bytes, that would be 1,500 minutes.

And the problem is, these things do rekey, typically every hour. Every 3,600 seconds an access point will rekey just as part of its security. So that's 60 minutes. So you don't have 1,500 minutes in order to be able to sit there and march down, being punished for minute every time you guess a byte correctly. So that limits you to the length of packets that you can apply this attack to effectively.

When you finally figure out what the MIC key is, and what the key stream is, that allows you to synthesize a packet up to the length that you caught and inject it up to seven times, given that that system has Quality of Service available, in which case it will not reject the packet as being a replay. Now, once you've done that, that is, you've injected that packet seven times, you can capture another one. And this time it only takes you between four and five minutes because you only need to get - now, if you capture another packet, it's going to have a different key stream, but it's going to have the same MIC key. So all you need is to get the last four bytes of the Integrity Check Value, and that only takes four to five minutes. You don't have to go the 12 to 13 minutes to get all of the last 12 bytes. You only need the last four. So that's the nature of what these guys have done. It means that...

Leo: I guess one of the takeaways I have is that, first of all, it has to be a router with QoS.

Steve: Yes. And if you...

Leo: Has to be enabled.

Steve: Yeah, exactly.

Leo: And they don't really - they don't crack your content. They can inject stuff into it.

Steve: Yes. Well, they do crack your content, but not probably any very valuable content. They can crack some of the management of your network, your network management. Now...

Leo: But they can't sit there and watch what you're doing online.

Steve: Correct. Correct. They never get your key. They're never able to get all your traffic.

Leo: So what are they able to do? I mean, what good is this?

Steve: Well, and that's one of the reasons why, you know, people saying, oh my god, WPA WiFi security is cracked, head for the hills. I mean, it's like, okay. I mean, this is a - now, admittedly, this is the way these things begin. So this is like the first chink in the armor. This is a wedge into something that we thought was completely secure. Not so much.

Leo: But is there reason to think that you could go to the next step? It's still a pretty big step to go to the next step to decrypting every packet. Since they aren't getting the key.

Steve: It is a huge big step. I mean, yes. No one knows how to do that. They're not getting the key. They're getting one key stream, and they're able to generate a few packets from it. Now, there were a few interesting things. Because, first of all, they're also only able to intercept from the access point towards a client, grab one of those packets. So they're only able to access in one direction because of their need to use this MIC failure frame in order to determine whether they guessed right or not. So you have to have quality...

Leo: It really sounds like a surprisingly limited hack.

Steve: It really is, Leo. I mean, it is uncomfortable, but it's very limited. I mean, and, okay, so maybe, maybe you could send a spoofed ARP, do like an ARP spoof and cause a client to send its traffic to a different IP. For example, and we've discussed this in the

dark ages of Security Now!, if you sent - if you have an encrypted client, and you get it to send its traffic to a different IP, then the access point decrypts it, and then it goes out onto the Internet in the clear, bound for the wrong place. Except that ARP spoofing doesn't really let you do that. ARP spoofing would allow you to send it to the wrong MAC address. So if you went to the wrong MAC address, if the client didn't know how to decrypt it, and it wouldn't, then that doesn't help you, either.

So these guys haven't told us anything that you can do bad with this. They've sort of said, well, you could decrypt ARP packets, or maybe DNS, like small packets. They're locked into small packets, and they're very much locked into what you could actually do. We're going to have to wait a while, probably, to see if anyone comes up with something clever that you could actually do with this. There may not be anything that you can actually usefully do with this. I mean, this...

Leo: You know, they're giving this presentation today or tomorrow.

Steve: Yeah, tomorrow afternoon, that's Thursday.

Leo: Might they reveal more? Or is this the full story?

Steve: This is it. They've published the paper, which discusses this, basically. There is code now in Aircrack. Martin Beck is an Aircrack contributor, so there is code in the existing Linux build of Aircrack, which Martin put in for Erik because they were working on this together in order to sort of test some of this, to see if they could actually do it. So they took it from theory to practice. They were able to decrypt the one packet. They were able to use Quality of Service channels to get that packet not rejected, but accepted. But they haven't been able to do anything more.

Leo: Which is not that surprising. I mean, they haven't - what is surprising is how it was treated as...

Steve: As the end of WPA.

Leo: Yeah.

Steve: As the end of WiFi encryption. I know.

Leo: I mean, I have to say now, knowing this, I'm not that afraid of somebody sitting out on my porch trying to crack my TKIP. I mean, they aren't really cracking it.

Steve: No. They're not cracking it.

Leo: [Indiscernible].

Steve: Well, and that's per session, and that changes every hour. So even if they get the MIC, they still don't know the key. And all the MIC does is allow them to do succeeding packets in four to five minutes, rather than 12 to 13 minutes.

Leo: Can they mess with me? I mean, could they, like, send a fake - I'm trying to think of a heinous application for this. Can they send a fake packet that - I don't know. They don't know the key, so they can't encrypt a packet, so they can't really - they can do ARP spoofing because they can spoof the IP address. But they can't spoof a packet.

Steve: Correct.

Leo: So they really don't - they ain't got much.

Steve: They really don't have much. And if there's one thing worth mentioning is that new access points will be offering - anything that is WPA certified, I'm sorry, WPA2 certified, will, in order to get that certification, have to be able to both do TKIP, and we'll call it AES, although it's really CCMP, either of those protocols, CCMP being the really secure one, modern, with no compromises to the past, that uses the 128-bit AES Rijndael cipher.

Leo: But before we get into what people can do to mitigate this admittedly not so horrible hack, let's take a break. But we're going to come back, and you're going to explain what settings you should set on your router, what you should do in response.

Steve: Yes. There are things, there are some things that are still dangerous, and some things people can do.

[Commercial break]

Leo: All right. Let's get back to this because now any time you see a headline that says "WPA Hacked," the first thing I want to do is go out and tell my listeners, well, what do you do to fix it. Now, you've actually, if anybody who's really understood what you're talking about, you've actually - well, I mean, look, I'm not going to talk about this on my radio show. It's way too complicated. But you've actually mitigated it somewhat by showing what's possible. It's not a lot.

Steve: No. It's not a lot. We'll see what clever people are able to come up with. But at this point it's sort of a technical hack. So here's the issue. Anybody with updated firmware, who's bought a WPA2 router, maybe a WPA-certified access point in the last few years, they'll probably have both TKIP and, for the sake of agreeing with the UI on the router, I'll say AES.

Leo: Yeah. It always says AES. But it's CCMP is really the encryption?

Steve: Yes. Exactly. CCMP is the protocol the way TKIP is a protocol.

Leo: Yeah, all right.

Steve: So the problem is, if you enable both on the access point, TKIP is still present.

Leo: Got it.

Steve: And it's still in the air because the broadcasts that are sent out to, for example, ARP broadcasts, they need to be sent out to the lowest common denominator so that the access point knows that everybody will be able to receive it. So the only way, if you are able to disable TKIP, the only way to know that it's not in the air is to remove it from your access point. Do not include its support in your access point along with AES, CCMP encryption.

Leo: So uncheck it or...

Steve: Yeah, exactly. Often you'll have, like, radio boxes where it'll say TKIP, AES, or TKIP plus AES, for example.

Leo: Ah, okay.

Steve: And so you do not want to use that. You don't want TKIP anywhere near your system. Now, okay. We've all said, okay, this doesn't look like it's such a big deal. The point is, fine, but why use it at all if you don't need it.

Leo: Right.

Steve: Now, we know that there are poor people - you know, sad people - who are even...

Leo: Unfortunates.

Steve: Unfortunate, that's the word I was looking for, unfortunate people who are still having to use WEP in some cases because they've got some refrigerator or TiVo or something, some device that is mission-critical to them...

Leo: I was so miffed, I bought this Rovio robot that's supposed to go around and take pictures. It uses WEP. It's like, what are you thinking?

Steve: As long as you don't mind someone else commandeering it and driving it out of the house and down the street to them.

Leo: Yeah, right, no big deal.

Steve: Yeah. So...

Leo: So there are some unfortunates that have to use devices like that, or Nintendo DS is another very [indiscernible] example of that.

Steve: It may turn out that there are similar situations where you've got something that is only TKIP. You might, for example, have some equipment which was only WPA certified and did not include the gratuitous AES, or included a non-final compatible version of AES, so that if you switch your access point to your WPA2 access point, that really supports real AES, if you disable TKIP, you may not be able to connect. I would say, because there's still some unknowns about this, if you don't need TKIP, disable it. You just change your access point to turn it off. You probably don't. The majority of people who are using Macs and PCs and Linux machines, and anything in the last few years will support AES and TKIP, it's better just to say, okay, no more TKIP. If you have to have it, you don't have to worry that much. If you can, you have the option of turning off QoS, disable it because that will probably defeat this attack also, if you needed to keep TKIP. And turning off QoS won't have any effect because most things don't use it anyway.

Leo: We use - I use it for Skype. I use it for Skype. So...

Steve: Okay, you actually do?

Leo: Yeah. But I'm not using it - I'm using it on a wired router, so I don't care.

Steve: Okay, but how do you know - so is Skype smart about saying that I want to use a different QoS?

Leo: Well, I don't think so. What I do is I tell the router that stuff that's coming in on this port, turn on QoS. Is that not enough?

Steve: Well, it might prioritize the port traffic. But normally...

Leo: Good point. Doesn't help between you and me, the part we care about.

Steve: It's normally the case that the packets themselves have to be marked, they have to have an 802.11 QoS header that says I'm to be given higher priority.

Leo: And then all the routers across the internet have to support it, which of course none do, probably.

Steve: Or at least it needs to get out of your point of congestion. But you're right, once it gets out of the 'Net, lord only knows what's happening.

Leo: Right. Well, and I don't, you know, because one of the reasons we get such good Skype results is I have a dedicated Internet connection with nothing else on that connection, and nothing else on the router, so it's completely pointless.

Steve: Right.

Leo: When I was upstairs, we had everything on one connection. And maybe it made sense then.

Steve: Right. So bottom line is...

Leo: You can turn it off in most cases, in other words.

Steve: I think most people, 99.99 percent of the people will be able to turn it off. When your friends come over, they'll have a computer from this century. And so it'll probably work just fine on your router with AES. And you do want to disable TKIP because, if it's on, it's out in the air, and you're still vulnerable. But again, if in the worst case you need it for whatever reason, it's probably not such a big deal. This is nothing like key recovery where they can now intercept your traffic and monitor what you're doing. I mean, it's just - it's one small packet they're able to get and decrypt every four to five minutes. And you're not sending small packets. Your payload is 1,500 bytes. That they'd never have time to decrypt because your system rekeys every hour. They don't have 1,500 minutes. They've got 60.

Leo: So to summarize, the easiest thing is probably to turn off QoS. None of this works unless QoS is - your router supports it, and it's enabled.

Steve: Because the modified packet would have the same counter value, and it would just be rejected as a replay. And that replay protection absolutely does work. It was them cleverly realizing that QoS allowed them a way around that that made all of this possible anyway.

Leo: If for some reason you need or want QoS - and if your router doesn't support it, then you're home free anyway.

Steve: True.

Leo: So your router has to support it. If for some reason you want to continue to use it, the next - the other thing you could do, and there's no reason not to do this, either, is to - you have to go to WPA2 to use AES. Is that correct?

Steve: No.

Leo: No.

Steve: Remember, because WPA is not a place you go to.

Leo: Understand. Understand.

Steve: It's just a certification.

Leo: Right.

Steve: So it is the case that...

Leo: The router is either - if your router is WPA2 certified, then you're fine.

Steve: You definitely - then you know you have a final standard AES protocol, that is, the CCMP protocol. A non-WPA2-certified router, that is, WPA, it may work. You know, just try it. Just switch to WPA and see if Windows and Mac and Linux still connect up to it. Because they may have gotten it right.

Leo: But so you turn off that TKIP, turn on AES. You'll be using that safer CCMP or whatever it is encryption...

Steve: Protocol.

Leo: ...protocol using the Rijndael encryption. And you don't have - and then you're completely in the clear, too.

Steve: This whole thing is then of no concern to you.

Leo: Is it, do you think, worth doing that anyway? Is it a better - is AES preferable anyway?

Steve: I would say that the fact that this happened is making people nervous. It's sort of the way we found some collisions with the smaller hash functions. It's like, they're not broken, but we're a little uncomfortable now that we found some problems.

Leo: So if you have a WPA2-certified router, or even if you don't, turn off TKIP, turn on AES, doesn't hurt.

Steve: Work just fine.

Leo: And nowadays everything, I mean, yes, it's a longer key. It requires more CPU power. But everything's fast enough. Is that CPU power, it's both in the router and in the connecting machine have to do it; right?

Steve: You know, one of the reasons that Rijndael was chosen as the AES standard is that it is very efficient computationally. It is a computationally efficient cipher. So it is - and it's a symmetric cipher, and we know that those are generally much faster than asymmetric ciphers. So it was mostly to preserve hardware that - where the RC4 pseudorandom sequence generator and that ICV tack-on was just - it was built into the hardware. There was nothing they could do to change that. So they said, okay, we're going to change the interior of the packet so the hardware could still generate the same kind of random numbers, and it could still tack now not very useful four-byte CRC32 on the end. And so it was mostly a - it was so that, you know, backward-compatible hardware could still be used in a secure fashion.

Leo: Yeah. Very good. You know, I can understand why mainstream media might have gotten this wrong. But to overstate the risk is really unacceptable.

Steve: It was, yeah, I mean, people were freaked out and panicked needlessly. And I was sort of smiling to myself when you said you could understand why mainstream media got it wrong. You can understand how hard it would have been for them to get it right.

Leo: Yeah. But...

Steve: It took this podcast to explain it.

Leo: But I think that they should have not jumped to the conclusion they jumped to, if they didn't understand it. In other words, if you don't get it, don't just assume that

we've got a crack here.

Steve: I mean, yeah. I mean, the headlines were saying, "WPA WiFi Is Cracked." You know, your network is no longer safe. It's like...

Leo: Is that, do you think, because Erik Tews might have overstated it?

Steve: No. I think it's because it is really complex. I mean, okay, Leo...

Leo: And it's also a sensational story that you're going to get...

Steve: I have two WiFi routers here, one on my internal network, one on my cable modem. I haven't changed either of them.

Leo: You don't care.

Steve: Well, I mean, this is not a big problem.

Leo: As Zephyr is saying in our chat, "WPA Almost Cracked" is not as big a headline. It's not as...

Steve: As they say, you're not going to sell so many newspapers.

Leo: No. Doesn't really grab you. Oh, it was close.

Steve: Or a "Partial Long-Duration Replay Attack," that's not going to sell anything either.

Leo: No. In fact, I don't think I'd click on that link. All right. Very good. As usual, Steve Gibson, you are an asset and a boon to the community because you can figure this stuff out and explain it, I have to say, explain it in a way that makes perfect sense, doesn't require a Ph.D. to understand, and put us all, our minds at rest. I still am going to turn on AES and turn off TKIP. But what...

Steve: Yes. And I should say, I should also say I haven't done so just because I haven't done so yet. It's not - I don't see it as, like, a critical emergency. I will, next time I'm visiting my wireless routers, I will switch over.

Leo: Next time you log in, flip that switch. Steve's at GRC.com. That's the place to

go for his fantastic software, SpinRite. Just, what is it, 62K of Assembly language goodness. It's just the program you want if you have a hard drive, and you need to maintain it and/or restore it. I use it all the time, on every drive before we install them. And I recommend you do the same.

Steve: You know, I ought to mention that while people are in their routers and reconfiguring them, if they don't know about GRC's Passwords page, that's a great place to get a password. About 3,200 people use it every single day, day in, day out. It's just GRC.com/passwords. And an extremely high-quality, extremely random gibberish password is just presented for you, over a secure connection. It never repeats. I've got all kinds of code to make sure that the same thing is never issued to two people anywhere in the galaxy, so.

Leo: And in fact that is a good point because WPA is a little vulnerable to brute-force attack, and that...

Steve: Oh, yes, the only known attack against WPA - well, okay, here we're using the wrong acronym - against the AES CCMP encryption, which is part of the WPA certification, is a brute force, where they guess what your key is and try using it against a packet which had been captured. That's an offline attack. It's still not a big problem, as long as you use a really hard-to-guess password.

Leo: Well, yeah. And these passwords at GRC.com/passwords are as random and as long as you can get.

Steve: Yup.

Leo: The best passwords you can use. The only issue is remembering them. But you don't have to remember them. You just save them somewhere.

Steve: You can't. Yeah, you can't even type it into stupid Apple iPod Touch because they're just so long.

Leo: Yeah, you have to cut and paste. And if you can't cut and paste, then don't do it. All right, Steve. GRC.com is also the place to get the 16KB versions of this, for those of you who are bandwidth impaired. Share them with your friends, they're tiny. Also of course the transcripts from Elaine. And this one is another one you might want to read along while Steve talks. This should be a graduate-level course in security, frankly. I mean, this is - you've got your textbook right there. You can also find all these great programs like the Perfect Passwords, ShieldsUP! for testing your router, lots of great stuff. GRC, Gibson Research Corporation. Steve, we'll talk again next week.

Steve: Talk to you then, Leo. Thanks.

Leo: Bye bye.