Listener Feedback #24

Description: Steve and Leo discuss questions and comments from listeners of previous episodes. They tie up loose ends, explore a wide range of topics that are too small to fill their own episode, clarify any confusion from previous installments, and present real world 'application notes' for any of the security technologies and issues they have previously discussed.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-110.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-110-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 110 for September 20, 2007: Listener Feedback 24.

It's time for Security Now!, everybody's favorite security podcast, in fact the number one tech podcast in all the nation, according to the Podcast Awards. Congratulations, Steve Gibson.

Steve Gibson: Thanks to our listeners, yes.

Leo: You're getting your award next week.

Steve: Yes, in fact I have a note here for next week's podcast to mention that I'll be picking up the award the following day after that podcast comes out. And I had a note here to thank our listeners again, so you've given me a perfect opportunity to do so, Leo.

Leo: We'll just keep thanking them until next year, when you'll win again.

Steve: That's my plan, as a matter of fact.

Leo: So if you're going to be in Ontario, California, on the 28th is it, yeah, of September, go to the Podcast Awards. They're usually towards the end of the day, so it'll be 5:00, I think.

Steve: Yeah, it's between 4:00 and 5:30, they said. And for what it's worth, the show is open and free to the public for the random walk around the aisles and see what's going on in podcasting land.

Leo: Yeah, so if you want to learn about how to podcast, this is the place to do it. Everybody's there.

Steve: Yup.

Leo: So - I didn't mean to scare you.

Steve: Agh.

Leo: It's definitely a great - I will not be there. I'm going to a different expo the following month, and I have actually commitments on the 27th in Vancouver, so I can't get to Ontario. But I wish you the best. I'm sorry I can't be there to applaud you. But I'll be there in spirit.

Steve: Ah, well. You and I have done it the prior two, that is, the first two of those.

Leo: Yeah, yeah. When I won, you were there. So, well, I think we're both winning for this, so I feel good about it.

Steve: Oh, yeah. This wouldn't be here, this wouldn't be happening, Leo, were it not for you, so...

Leo: I'll take a little...

Steve: That fact never escapes me.

Leo: I'll take a little credit.

Steve: Well, and vice versa, I suppose.

Leo: Yeah, I think you should take a lot of credit. So this is a listener feedback episode. We've got a lot, 13 questions, 12 good ones and one wacky one.

Steve: Yeah, the wacky one, we're getting such fun feedback. I had a hard time literally pulling myself away from my email client this morning when I was going through these. I just - I did not want to stop reading them. But we got one that was just so bizarro, I thought, well, it's perfect that it'll be number 13. So it'll be the wacky 13th one.

We have a bunch of errata, well, several bits of errata that I wanted to share with everyone. First of all, I wanted to just do a shout-out, as I think they use the term, to the two little bots on Mars, Spirit and Opportunity.

Leo: Oh, tell me about this. I guess I haven't been following it. What's going on?

Steve: Well, they had what was originally planned to be a three-month mission. They are now both in their 43rd month.

Leo: Holy cow.

Steve: They are still going strong.

Leo: Oh, I remember these guys, yeah. They weren't supposed to last very long.

Steve: Right. Spirit and Opportunity were these two bots. And, I mean, they landed literally 43 months ago, bounced around in their little balloon landing pods, and then unfolded and came to life. And they've just been doing fantastic science for far longer than anyone expected them to last. There was a big global dust storm last month which obscured the sun and brought the power levels that they were able to obtain through their solar cells down to a very low level. And the guys at JPL were worried that that might do them in. So they put them to sleep into a very, very low power mode, where they stopped doing any science, they just sat there, and they only sort of pinged their radios as little as necessary.

Well, the storm passed, they did have a bunch of sand on their solar panels, but some wind picked up and blew that off. And they just took off rolling around again. It's just so cool. You know, just these two little bots roaming around Mars, just continuing to go far, far longer than they were expected to last. And, I mean, there are all kinds of - like there was a long wish list of things that NASA wanted, if they had their dreams, these things would be able to be done. And the bots are just tackling them one after the other, just sort of cruising around.

Leo: Yeah. Well, good for them.

Steve: That's just so cool.

Leo: Good for them. I'm proud of them.

Steve: Yeah. So I just wanted to give them a little shout-out. I'll kind of keep my eye on them and let our listeners know if they continue trucking along or if something finally - I mean, I would think that, if they fell into a hole or something or, you know, got tipped over, that'd be the end of the story. But they're just - they're doing a great job.

Leo: That's called over-engineering, and it's the way to go. I think it's fantastic. That's really great.

Steve: Many, many, many, many, many listeners...

Leo: Many.

Steve: Many commented that the idea that I talked about last week that I had

come up with for storing the state of eCommerce transactions in a variable that I gave back to the browser each time was something that Microsoft has offered in their ASP.NET suite for a long time. There was an article in 2002 that I found, just to give it some sort of date reference. So, and this is called View State. So I just wanted to acknowledge all of the many, many, many, many, many, many listeners who said, hey, Steve, cool idea, but you didn't invent it. Well...

Leo: I don't think you claimed to invent it.

Steve: No.

Leo: It's the idea behind cookies in general.

Steve: Well, exactly. Or behind saving state. And I just wanted to talk about the solution I found. Several other people wrote, in fact I have - one of our Q&A guys I have a comment about his approach. So we'll talk about it a little bit more. But I just wanted to acknowledge that, yes, I didn't intend to claim that I had invented, it's just, I mean, I invented it for myself. But it's sort of the nature of the question of what's invention and what's engineering, which is why patents are so questionable, is some things are just the way you solve the problem if you're schooled in the art. Anyway, so I wanted to acknowledge all of those people.

Also I mentioned when we were talking about the Personal Identity Protection, the PIP program offered by VeriSign through their VeriSign Identity Program, VIP, I talked about how it was my hope that at some point in the future the wacky URL that we had right now, like steve.gibson.pip.verisignlabs.com, would be made shorter. Well, the technical guy at VeriSign listens to the podcast, or at least he does now. And so he dropped me a note saying, hey, Steve, I wanted to make sure you knew that the OpenID system allows something called "delegation." And I had originally talked about that when we discussed OpenID. And the idea was that a user, that is, an end-user who wanted to use OpenID could put on any web page that they control, that is, their own blog page, their own MySpace page or whatever, any page that they have control over, they're able to put a tag on the page such that they refer to their own page as their identity, and then the server pulls that page, looks for the tag, and then uses the contents of the tag as the target for their identity server. And so the point of that is that, in fact, I could use something like GRC.com to replace that entire long nightmare, steve.gibson.pip.verisignlabs.com. So I could log onto an OpenID site just using GRC.com.

Leo: That's the whole idea of OpenID.

Steve: Exactly. And then that site would go to GRC.com, look for the tag, find buried in the tag

steve.gibson.pip.verisignlabs.com, and then go there in order to start the validation process.

Leo: Yeah. That's what I do. I use Leoville.com as my base OpenID page.

Steve: Well...

Leo: The code [indiscernible] sitting there. If I had thought of it, I would have mentioned it. That's kind of the premise of OpenID is, if you have a site, you can do that.

Steve: Exactly.

Leo: If you don't have a site, you can't do it.

Steve: And then we have a listener in Toronto, or who has returned to Toronto, Michael Walker. He says he was recently flying back to Toronto from London. And he wrote, "I thought that you would like to know that while on an Air Canada flight from London, England, back to Toronto, there was an ad for Nerds On Site."

Leo: Oh, that's neat.

Steve: "It was at least a full minute ad on the in-flight television." And so he says, "Well, on the screen you can see the nerd working on the computer. But on the computer screen, and very clearly seen, is..." and he puts in parens "(wait for it...) SpinRite." Way too cool.

Leo: Ah, that's neat.

Steve: So, yeah, I really got a kick out of that. Apparently the Nerds On Site guys have an ad that runs on Air Canada's in-flight TV. And they figured, well, what are we going to put on the screen? Let's put SpinRite on the screen.

Leo: Well, they're proud of the fact that they offer a license to SpinRite, so...

Steve: Yeah, exactly.

Leo: So we've got 13 questions. I suppose we should get started. Do you have a SpinRite anecdote you'd like to recount, or...

Steve: Actually I do. I found one from a month ago that sort of follows on what we talked about last week. Remember that last week we had a listener who wrote in saying that he had used SpinRite in Wine, under Linux, on the Windows emulating...

Leo: Oh, yeah, and it worked.

Steve: And it worked. Well, this has been taken to an extreme now by Eric Greve, who wrote on August 21st, he says, "Hi, Steve and Leo. I've been a SpinRite user for quite a while, and it has saved my bacon more than once. Whenever I buy a new hard drive I plug it in and let SpinRite have at it. I seem to be addicted to buying hard drives, so this happens more often than you'd think."

Leo: That's good, though. I mean, that's who should have SpinRite. Anybody who uses a lot of hard drives, that's really where you get your value out of it.

Steve: Well, yes. And in fact, I mean, maybe it's a holdover from the old days when I used to buy hard drives, and the first thing I would do was run SpinRite because back then hard drives were dumb.

Leo: You had to, yeah.

Steve: They didn't have all this brain power that they do now. Which I think, you know, moderates the need to run SpinRite on a brand new drive...

Leo: Steve.

Steve: ...although I do.

Leo: This is your business here.

Steve: I do.

Leo: Don't tell them they don't need it.

Steve: So he says, "I just bought a new drive again, and I thought I'd try something new. You see, I've also been experimenting with virtualization using the free VMware server on Linux. While looking around the options, I saw that I could give full control of a physical hard drive to a virtual machine. They warn that this is advanced" - ooh, watch out - "and maybe dangerous, since both virtual and real OSes could use the drive at the same time if misconfigured." He says, "I figured I'd give this a try with SpinRite." And I love what - he went all the way. He says, "So I configured a new virtual machine, gave it the SpinRite ISO file..."

Leo: Oh, there's a good idea.

Steve: Isn't that cool? "...as its CD drive image..."

Leo: So it's tied in.

Steve: "...and full control of the new drive as its sole hard drive..." and, he says in parens, "(the other ones being all in use by the Linux system)..."

Leo: This means that I could use it on the Mac.

Steve: Yes, actually.

Leo: I like this.

Steve: Then he says, "...making sure to turn off all caching and snapshotting." He says, "I'm happy to report SpinRite worked flawlessly in its own little world as I was using the real system normally."

Leo: Wow.

Steve: "I just wanted to..."

Leo: That's a great solution.

Steve: Isn't that cool? He said, "I just wanted to let you know about this maybe unusual use, and ask if you knew of any caveats with using this method for testing my drive. Thanks for the great software and podcast. Eric from Montreal, Quebec, Canada." So you're right, Leo, I mean, it does mean that in VMware, and remember that the VMware server is free, that...

Leo: On PCs.

Steve: Exactly, for PCs. And what about for Linux?

Leo: No. Yeah, maybe it is for Linux, yeah.

Steve: I would imagine it would be.

Leo: Not for Mac, unfortunately.

Steve: Anyway, you can run SpinRite in a VM. And the other cool thing about this is...

Leo: It's a great idea.

Steve: ... SpinRite is an assembler, and all it's really doing is data transfer. I mean, it's not making intensive use of the processor. Most of the time it sits around doing nothing.

Leo: It's the controller, really, that's doing all the work.

Steve: Exactly. So you're issuing a read or a write. So that means that it's not going to suck up a lot of your machine if you want to be running SpinRite on a drive that you've given it exclusive access to, sort of in the background. I mean, it's not something that would slow down your foreground use because SpinRite's not sucking up much of your processor.

Leo: Yeah. Is the only call you make to N13? Is there still an N13?

Steve: Yes.

Leo: Is that what you use?

Steve: Yes. I use the N13 BIOS, which is - and the BIOS is fully emulated by VMware. In fact, you can sort of see the little BIOS flash screen start up, or splash screen start up when you initiate a virtual machine. And the other cool thing is that, since this is just DOS, it doesn't need gigabytes of RAM. In fact, it doesn't even need megabytes of RAM.

Leo: It needs kilobytes.

Steve: Yes. You could give it 640K, a 640K virtual machine, and it would be just, I mean, it would be absolutely happy running in a 640K virtual machine. So it wouldn't even take up much of your RAM, I mean, less than an app would.

Leo: Oh, this is really good. So I'm going to make - what I'll do is I'll make a VMware virtual machine that's a SpinRite machine. Obviously you can't distribute it because it's a commercial program. But I can keep that and use that whenever I need to check a drive. Just pop the drive in and boom, there I go. Wow, that's great.

Steve: Exactly, give the drive...

Leo: It has to be full access, yeah, right.

Steve: Yup, in order to have full physical direct access to it. But then, you know, it'll work.

Leo: Very cool. Coolness personified, impersonated, I don't know what that is.

Steve: Whatever we're talking about.

Leo: Whatever it is. All right, let's get to our questions. I have them in front of me in my hot little hand, 13 good questions for you, Steve. Are you ready?

Steve: Well, 12 good ones and one wacky one, so.

Leo: You got your thinking cap on? Starting with Kevin in California, he's asking about why we don't do Security Now! in other audio formats: I've been listening to Security Now! for about a year. I was wondering if you were considering making an Ogg Vorbis version of it. He uses open source software every day, and he's trying to distance himself from proprietary formats like MP3 or WMA. He points out that Ogg is unpatented, royalty free, delivers superior quality to MP3. More info and encoders are available at Xiph.org. Thanks for your time. I get this question all the time.

Steve: And so I wanted to voice it and have you answer it for us.

Leo: We do, you know, the only podcast - we do two podcasts in multiple formats. One is TWiT, which for historic reasons - and I deeply regret these, by the way - we do high, medium, and low MP3; we do an AAC version; we do an Ogg version. And it's a pain in the butt. It actually adds - probably doubles the amount of time it takes to produce the podcast. And I'm frankly thinking of phasing those out. We also do - now, FLOSS we do in two versions. One is MP3, and one is Ogg. And the reason we do Ogg is because many of our listeners on FLOSS Weekly are Linux users. And they can't use an MP3 player without signing a license, which many of them are not willing to do. So we do - and maddog pointed this out. So we do an Ogg version of it. I really don't want to - the reason we use MP3 is every player plays it. Very few players play Ogg. Almost always you're going to be playing it on your computer. And it just adds considerably - we're already producing a high and low version, and ad-free versions we produce of this show. And it's just - it adds considerably to our production costs and time. So...

Steve: That makes total sense, Leo.

Leo: When I look at the number of downloads for the Ogg versions of other podcasts, it's in the hundreds. I mean, TWiT, which has something like 150,000 downloads a week, there's hundreds of Ogg downloads. It's just a lot of work for a very small number of people.

Steve: Very little return on the amount of work it would take. And also, as you said, most of what people are doing I'm sure is not listening to these things on their computer, although certainly some are, but the vast majority of people are using podcasting in the original iPod podcasting mode, where they dock their hardware, and it downloads into the hardware, and all the hardware out there plays MP3s.

Leo: Right. So I chose the lingua franca. Kevin, I understand what you're saying, and you're right. I would prefer - I'm not a fan of proprietary formats. And MP3 is technically a proprietary format, although it's in such wide use that it's kind of - that's kind of moot. I just can't add the extra time. And I know he's going to say, oh, you can write a script, and you can automatically do it. In fact, I have scripts to do it. But then they have to be tagged, and that has to be done by hand, and they have to be uploaded, and we have to

find storage for it. And it's just, I'm sorry, it's just not going to happen. In fact, I wasn't doing a low-quality version until Steve asked us for a dialup version, and I agree. Almost all the other podcasts are one version only, MP3 at 64 kilobits.

Steve: And so it is the case that all these players that are playing MP3 format, they're sending a little royalty check to the MP3 gods somewhere?

Leo: Well, the rules are complicated and unknown. Fraunhofer owns the - and Fraunhofer was acquired, I think by Lucent, and it's very complicated. They invented it, and they won the rights to it. And if you make - I can't remember what the rules are. There's a certain length and blah blah blah, and it goes on and on and on. So it's complicated. It is encumbered. It is not a free and open, license-free - I think, for instance, Winamp, for instance, probably pays an MP3 license. And that's why you see a lot of software that's for audio, I think, doesn't support MP3 directly. So...

Steve: Interesting.

Leo: Yeah. It's kind of people act as if it's open, and it's not. So if you want to read all about it, go to mp3licensing.com. Thomson now owns the license to it. You can read all about the cost of it. And in fact the codec is a buck 25 a unit, or $90,000 one time only for the decoder. So, but I don't - but I think there are rules about who has to pay for it and so forth.

Steve: Interesting.

Leo: Ultimately we may be forced into a situation where we have to stop using MP3. You know, the right to distribute MP3-encoded data has another license. And...

Steve: But these are, okay, these must be patents; right?

Leo: Yes.

Steve: And so they're not going to last forever.

Leo: Right, thank goodness.

Steve: So just like the old Lempel-Ziv compression, where, you know, I mean, that's now...

Leo: That's right, yeah.

Steve: That's now in the public domain. So it seems to me at some point, probably before this gets any worse, it'll end up being finally free. And then Linux people will be able to get MP3 files and players and everything.

Leo: Yeah. Well, and they can now. So it's more about purism than it is anything else.

Steve: Yeah.

Leo: And I'm sorry, I just have to be pragmatic. And I understand, Kevin, I completely understand. I get this request fairly frequently. But when I look at the numbers, not that many - truth is, you know, a lot of people talk the talk, but very few walk the walk. It's very few people who actually download the Ogg versions.

Question 2, an anonymous listener complains, comments, that GRC's redesigned feedback form is not screen reader friendly. What about that? Boy, this is complaint central today.

Steve: Well, yes. I wanted to respond to that because he's right. But unfortunately he's right by design. That is to say, when I took the form off the bottom of the Security Now! page and gave it its own location, and just to remind our listeners it's GRC.com/feedback, what I was trying to do was to moderately obscure it from spambots. We had this problem where that page had attracted some spambots, and it had begun getting spammed just by bots that were filling in the form. My guess is that nobody deliberately targeted some specific bot at the form. That is, I don't think a bot was designed or scripted to send me spam. I wasn't getting that much. But what must happen is that there are just bots out there, in the same way that search engines are, crawling around the web, looking for forms and going, oh, do I recognize the field names on this form? If so, they dump their spam in and push the button. And whoever it is who's reading it gets spammed, which is really annoying. And this may be a variation on, like, the blogging spambots that put spams on blogs.

So my sense was, okay, I don't think we're a high-value target. I didn't want to inconvenience our users by requiring them to do a CAPTCHA. So I figured, I'm going to just sort of make it less obvious that this is an email submission form by moving the labels of the fields from text into images. You know, not a big, super-secure thing to have done. But all the spam is gone for now, and it just solved the problem. And I'm not needing to ask our users to jump through hoops by figuring out, you know, a CAPTCHA every single time they want to submit something.

Leo: But if they're blind it doesn't - they can't see it with their screen reader.

Steve: And that's the problem. Well, and here's the point. He says it's not screen reader friendly, and that's precisely the point because to be screen reader friendly would be bot reader friendly.

Leo: So if you put alt tags in, for instance, which makes it screen reader friendly...

Steve: Exactly.

Leo: But then the bot could read it.

Steve: Exactly. The same things that the screen reader would key on in order to allow it to be friendly to them would make it similarly friendly to bots. And so, again, if we start getting spam on it, I'll do something else. But unfortunately I did have to move it away from being screen reader friendly because I wanted it to be moderately bot hostile.

Leo: There's always tradeoffs.

Steve: Yup.

Leo: Jim Wilson in Southeast Kentucky - and I'm sorry for our blind listeners, and I know we have quite a few. Is there any way around that? Can they send you an email?

Steve: They could certainly send us email, and it'll get sent to me. Or if there's any way for them to look at - I mean, I don't know, to have someone look at the form and tell them what the fields are, then once they know, then everything would work just fine.

Leo: It doesn't change.

Steve: Exactly. I'm not dynamically scrambling things up. I'm not changing anything else.

Leo: Well, tell us what the fields are just for now, so they can jot it down? Do you have a copy? Is it GRC.com/securitynow? No, it's just /feedback; right?

Steve: Yeah, yeah.

Leo: So let me - I'll read them out to you. I'll be your screen reader. So if you do it by tabbing, the first tab will be to the subject. Second tab will be to the body of your message. Third tab is your name. Fourth tab is location. Fifth tab is email. And then there's a button which you can read which says Send Us Your Thoughts.

Steve: Yeah.

Leo: So subject, body, name, location, and emails. Make a note of that, and when you go to GRC.com/feedback, you tab once, you go to subject; and you tab twice, the second time is body; name; location; email. That should work.

Steve: Perfect.

Leo: There you go. And it doesn't change. And we're sorry about that. There's a reason for it, though. It's not unconscious.

Jim Wilson in Southeast Kentucky found an eBay CAPTCHA cracker. It's an add-on for Firefox. Great. Great. I've been a Security Now! listener since around 77, and I just found this Firefox add-on I thought you might find interesting. It solves eBay's CAPTCHA issue. It's add-on 4381.

Steve: Yup. I checked it out, and it looks like it's legitimate. There's a word back from the Mozilla developers, who looked at it and sort of are vouching for it. They say, yes, you know, it

looks to us like this thing works. And...

Leo: What does it do?

Steve: I just got a kick out of it. It literally, when you go to log onto eBay, eBay now, along with asking for your eBay userID and password, gives you a CAPTCHA in order to make logging on more difficult. Well, somebody got annoyed by that...

Leo: Vladuz.

Steve: ... and they created an add-on that recognizes where you are and decodes the CAPTCHA on the fly and automatically fills in the CAPTCHA challenge box.

Leo: Wow. If they could do that in XUL, it mustn't be too hard. In fact, the developer says this is only a proof-of-concept of how insecure eBay's CAPTCHA is.

Steve: Yeah, I don't know, maybe eBay is doing something dumb like they're putting an easily de-obfuscated link that actually has the CAPTCHA code in the link. I mean, I don't know that it's actually reading and doing an OCR number on the image. There might be some other thing that eBay's doing where looking at the text of the page you're able to determine what the CAPTCHA equivalent text is. I mean, I didn't look at it that much. But I just thought that was a real kick that, you know, we've talked about CAPTCHA and cracking, and here's an add-on that just says, oh, I mean, you use Firefox, and it just fills it in for you.

Leo: I wonder how it works. That's amazing. I mean, they must - wow. I'll have to look at the source code at some point. Mike Shaver, as you say, from the Mozilla Foundation - and I know Mike, great guy - says we checked the source code, it's harmless. Doesn't do anything to you. eBay may want to fix its CAPTCHA.

Brian Clark in the U.K. had an interesting malicious JPEG question. He points out that we've talked about how a malformed JPEG can trigger a trojan horse, he says, and that malware would get squished when it's resized or otherwise manipulated. He says: Is there any way that in general use, resizing, et cetera, malware could evolve due to the random changing of the contents when resizing, or is it not that sort of compromise? I'm just wondering if malware or bad effects could come about in this way, or would files have to be purposefully grafted to behave as malware. Crafted, I think is what he meant, not grafted, to become malware. You can't - they're not going to evolve.

Steve: Exactly.

Leo: Not a genetic experiment.

Steve: I got a bit of a kick out of it because we've had some similar questions in the past where people were worried that something malicious could sort of happen by mistake.

Leo: Right.

Steve: And of course this is also a variation on something we discussed a couple weeks ago about this notion of malicious video uploads. And we talked about how, for example, in the process of any kind of transcoding from one format to another, or even changing the size of the file in order to make it a fixed bitrate or a fixed physical image size, any of that would just blast any malicious stuff into history. The same is certainly true with malicious JPEGs. First of all, remember that JPEGs are only capable of being malicious if there's a known and unpatched vulnerability in the JPEG interpreting code. And so, I mean, that has been the case in the past. It was possible briefly to do malicious JPEGs, to craft malicious JPEGs in Windows before that known vulnerability was patched, as it has been. But even so, it's just like with video, any resizing, reimaging, remunging of the JPEG would lose that because inherently the process would produce a valid JPEG, and none of that code would survive.

Now, the one exception might be if there was explicit tagging in the JPEG that would survive, the so-called metatags where you add extra information. If that were to persist through the image or reimaging of the JPEG, then you could imagine that, okay, that might not be changed by something that was just changing the resolution or the level of JPEG compression. So conceivably something could stay. But he's sort of wondering if it could evolve as a consequence of random changing. So the point I wanted to make there was that, I mean, okay, we've heard the example of a million monkeys all banging on keys randomly on typewriters, and one of them produces a Shakespearian work.

Leo: Yeah, but it's an infinite number of monkeys typing for an infinite amount of time.

Steve: Yeah, and so you've got to have an infinite amount of patience, too, and read a whole bunch of nonsense before you end up with Shakespeare...

Leo: A nearly infinite amount of garbage.

Steve: Yeah. So it's virtually impossible that that would happen. While not being absolutely impossible, it's just not the case that random bit migration would do anything useful. It would just - in the best case it would crash your system rather than do something malicious.

Leo: Yeah. You know, I think - my daughter's taking statistics this year in high school, AP statistics. I'm so proud of her. And one of the reasons I encouraged her to do it is because I think that there's a certain amount of - John Paulos calls it "innumeracy." It's the numeric form of illiteracy. There's a certain amount of misunderstanding of math and statistics and so forth that goes on. And I wanted her to understand it better. And this is an example, which is just because it could happen randomly, you have to understand the vast unlikely, you know, improbability of this. Highly improbable.

Steve: Well, and literally, in a block of code, one bit that is changed will...

Leo: Breaks it.

Steve: Yes, exactly, will break it. So literally a single bit. There might be a couple bits in a 4 kbit block of code, or a more reasonable code, size is going to be - 4 kbytes is 64,000 bits. You might have a couple bits you could change that would not collapse the code. But most of them absolutely have to be the way they are.

Leo: Kind of like saying I'm going to put a bunch of carbon, hydrogen, oxygen, and nitrogen molecules in this cocktail shaker and shake it really hard and maybe a monkey will pop out. It's just not - it's not that likely. Okay?

Steve: A guess. Yes, you're right. That's a good example.

Leo: Dennis Wright in the U.K. - I'm sorry. I don't want to - I'm not being harsh. I understand.

Steve: I don't mean to be laughing at Brian.

Leo: We're not laughing at Brian at all.

Steve: The monkey coming out of the cocktail shaker, that's a great visual, Leo.

Leo: Yeah, well, sometimes it takes a good visual to [indiscernible]. It's funny, I really think that this is the kind of thing they should be teaching in schools. It's kind of critical thinking, especially with numbers, because we get lied to a lot with graphs and polls and statistics. And the ability to look at that and kind of do some critical thinking and go, that doesn't make sense.

Steve: Well, and actually it is a known - it's a known fact that people are not very good about statistics.

Leo: No.

Steve: That is to say, that is, we don't think...

Leo: It's not intuitive.

Steve: Exactly. We do not have an intuition about probability because it's just not the way we were built. People, for example, assume that, if you toss a penny in the air - and we'll state that pennies have an even chance of coming up heads or tails, in fact there's a slight bias...

Leo: But that's due to the weight of the penny, it's not - yeah.

Steve: Exactly. So the idea, if you throw a penny up in the air, and you do the heads or tails thing, and like five times in a row it comes up heads, there's this intuitive sense, which is wrong, that we're owed some tails. Like, oh my god, if we had five...

Leo: [Indiscernible].

Steve: Yeah, exactly. If we have five heads in a row, then whoa, there's like this built up need for us to have some tails coming up. But it just isn't the case. In fact, it's extremely unlikely that you'll get five heads in a row. But it can happen. In fact, we know from binaryness that there's a chance of one in 32. So in fact it's less unlikely than you might think because one in 32 odds is not that bad.

Leo: No.

Steve: It's like three out of a hundred, essentially, it's close to that. So you're not owed tails just because you got a bunch of heads. But we sort of tend to think that you are.

Leo: We think [indiscernible], yeah. You hit a hot button, obviously, and I probably shouldn't belabor it. But I think that it's important. There's a lot of superstitious thinking that goes on because we just don't get this very well. And, you know, things like thinking you're going to win the lottery when you have a greater chance of being hit by lightning. You know, we don't understand that.

Steve: And then what of course is annoying is every so often someone does.

Leo: Well, somebody's always going to win it. That's right. But it ain't gonna be you. Sorry. The chances are very slim. It's what built Las Vegas, ladies and gentlemen.

Dennis Wright in the U.K. has been programming web forms, too, something like what you've been doing for your eCommerce. He says: I was listening to the fascinating episode on your eCommerce system. I think I may have beaten you to the session management scheme.

Steve: Dennis and the rest of the world, apparently.

Leo: I don't think - I'll have to listen to it again, but I don't think...

Steve: No.

Leo: ...you claimed that you invented this.

Steve: No, I just came up with a solution that worked for me.

Leo: Yeah. Almost all websites use some form of session preservation using cookies. I mean, that's just kind of common. Anyway, he says: As you were talking, I could see where you were heading because I'd been there myself. I'm an actuary - interesting. Now, here's a guy who understands statistics. But he's interested in IT. Some years ago he created an interactive online retirement modeling system for the place where he worked. I had the same issues. You can't assume cookies or JavaScript are available, and you didn't want an overcooked database solution. So he had the data items, which were few and not needed to be recorded permanently, shuttling back and forth between client and server using the query string and hidden form fields. Oh, that's a good way to do it. I skipped the

encryption because the whole thing was running under SSL. No doubt you'll tell me that was a mistake. No, that sounds sensible.

Steve: Well, I wanted to put this in here because there were so many people who mentioned the idea that I talked about last week not being unique to me. Apparently there are books on CGI programming that talk about maintaining state this way, too. And so...

Leo: Most programs have to solve this problem. State is important in a program.

Steve: Well, yes.

Leo: And in a client-server environment you have to figure out a way to do this.

Steve: Exactly. So there's storing cookies, formal standard HTTP web cookies on the browser. There's various ways you could use variables in JavaScript. Or you could go with standard straight HTML, which is what I did, and just return the contents in a hidden field in the form in order to pass the state back to the browser, which it then passes back to the server when the user adds some more information to that form and then resubmits it. And I did want to talk about his comment about not using encryption because it was running under SSL. And he says no doubt you'll tell me that was a mistake.

Well, what SSL of course did was, for Dennis in this case, was it prevented anyone from sniffing the transaction as it was passing back and forth on the line. My concern, and the reason I encrypted this blob of data and then signed it and made sure, was that the signing, of course, verified that it had not been altered because essentially I'm taking all of the state of my eCommerce transaction and hanging it, transporting it out of the server off to a client somewhere. Now, I'm assuming he's on my side, that is, you know, this is his eCommerce transaction, after all. And he wants it to work out well. I'm not giving it to some bad person. On the other hand, I don't know that I can trust this person not to screw around with my server.

So when I hand them back the blob, I don't want them to be able to, A, modify it in any way, not do any sort of like delayed replay. Replaying the data in the future is something that I also prevent by putting in a transaction count as part of the encrypted data to prevent any kind of a replay attack. Nor do I want them to be able to decrypt it, see what's going on, and figure out anything they might be able to do with it to mess with GRC or eCommerce or whatever. So because browsers might be caching those pages, even though I explicitly expire the pages and have "no caching" tags all over the place, it's theoretically possible. So, and the fact is there's no vulnerable data, other than the users' own data that they have submitted in the form, ever coming back to me because all I'm doing is carrying that through a couple pages. But, you know, I thought, what the heck, I might as well make sure this thing is locked down. So aware of the dangers, even though we also have an SSL connection which is enforced through all this, I thought, I just don't want anyone to screw with this. This is my blob, not their blob.

Leo: Not your blob. Hands off my blob.

Steve: So I'm going to encrypt my blob, and I'm going to digitally sign it, and I'm going to put serial numbers in it, and I'm going to do everything I can think of so that the blob I give you is your data, but I've made it mine. And all your browser knows to do is send it back just the way it is. And then I'm going to take a look at it and make sure that everything about it makes sense before I trust it. And that's what I do when the data comes back. I deblobbify it and then

look at it, make sure it all makes sense, and then proceed to process that next submission from the user. And it works.

Leo: Yeah. That makes perfect sense. I mean, I think probably any book you look in is going to talk about some way to do this. And cookies are obviously the way everybody uses, just because it's there.

Steve: Although apparently Microsoft does have this thing that they call State View, which is built into their ASP.NET system, that makes it very easy for programmers to carry state from page to page in a fashion very much like this.

Leo: But I'm thinking it's cookies; right? I mean, if you're storing something...

Steve: We have to be careful about that word, though, because cookies are sort of a reserved word. Cookies means a blob stored on the server that's associated with the site.

Leo: No, on the client, on the client.

Steve: I'm sorry, yeah, of course. Erase that. Stored on the client that's associated with the site. So a cookie is a specific thing. This is not a cookie. This is an opaque token which is sent back and is part of the page itself.

Leo: Yeah, okay. So it's not stored.

Steve: Exactly.

Leo: On the client side.

Steve: It is not stored on the client side. It's on the page.

Leo: Right. If it's stored client-side it's, you know, cookies were created by Netscape, and originally they were called PCSSIs, which is not very catchy. Client side, what does it stand for, Persistent Client Side State Information. And so anything stored on the client has state information. That's a cookie.

Steve: Right.

Leo: But I guess since you're - yours are not persistent. That's the difference. It doesn't persist through the session.

Steve: Well, and but there are also non-persistent cookies.

Leo: Right.

Steve: I think what the real difference is that I'm storing it in the page. So the page that comes back to the user, it's actually in the HTML is this blob which is a field in the form that I'm next asking them to fill out. And so it just, you know, I'm not having to keep any memory of this on the server at all. When I'm done with it, I send it all back to the user. If they want to continue or proceed with the transaction, they click on the button or fill out the form or do whatever, and then all of that state information comes back to me as part of the next stage in the transaction. So...

Leo: Right. It's a session - it's effectively what a session cookie does, which is it's not saved. It's just for the session.

Steve: Well, except that a session cookie is normally a token that does refer to state information saved on the server. So this session cookie comes back to the server. Then the server looks up what the - like the bulky session state that that cookie refers to. Instead, all of that data is what I'm sending back out to the browser, and it all comes back to me. So I have nothing - so it's not a pointer to data on the server. It's all the data.

Leo: Well, I mean, I've used session cookies in programming, and most programming languages support this kind of [indiscernible] you can [indiscernible] me. Essentially a lot of times it's just, okay - I just pressed a button or something, lost my audio. Lot of them it's just you're setting a date or a time, and it's programmatic, I mean, it's a hash of something.

Steve: Right.

Leo: That the program did. It's not like the server's doing a lot of work.

Steve: Right.

Leo: Joe Gajewski of Buffalo, New York is worried he's running on borrowed time. Steve writes these teasers, by the way. I just want to give you credit because they're really good. I love Security Now!, says Joe, and I've been listening since day one. I heard you say a few times SpinRite's going to work on TiVos. But I own a Dish Network DVR. Can I use SpinRite on my Dish hard drive? He says, you know, it's old, and he knows it's going to fail eventually. But he hasn't been able to figure out if the Dish drive can be replaced by anybody but the Dish Network. What makes it so difficult to clone this hard drive? I'm not concerned with extracting programs, just backing up the drive so, if and when it fails, I can restore it myself. Oh, interesting.

Steve: Well, it's a great question. I know that DirecTV's DVR is TiVo based. So DirecTV...

Leo: Actually, no, they have two. One is and one isn't.

Steve: Oh, okay. So...

Leo: They've gone to a company DVR as well.

Steve: Okay. Well, the original ones were.

Leo: Right.

Steve: So the older ones probably are. And that means that they're Linux based. I don't know anything about Dish Network's DVR. But I'm absolutely sure that SpinRite could run safely on the hard drive.

Leo: You're independent of the file system. You don't care what...

Steve: Exactly. Exactly. In fact, in the PowerPC TiVos, which are the ones I use and the older original ones, the bytes are swapped on the drive so that it just looks like gibberish to SpinRite. SpinRite just tackles it one sector at a time, you know, as fast as it can, and plows through the drive. And it's also able to talk to the drive's - the extra ATA or IDE data, you know, the SMART data and all that stuff. So it's able to extract that from the drive and monitor it while it's doing this work, regardless of how the drive is formatted, or even if it's not formatted at all, like we talked earlier about some guy who bought a new drive and runs SpinRite on it. Well, he doesn't even have to format the drive to run SpinRite on it. So it's certainly the case, if you're able to open up your Dish Network DVR and get at the guts inside and pull the drive out and mount it on a regular PC motherboard, you'll be able to run SpinRite with no trouble.

Leo: Interesting. You know, I don't know anything about how Dish does it. I did write a book on the TiVo. And they're basically standard drives with - at the file system level, at the data level there's encryption and blessing and so forth. But SpinRite's not going to be a problem, and you should be able to ghost it as long as you use something very low level that's just doing a sector copy of it.

Steve: Exactly. In fact, there are a bunch of Linux command that are very good about doing just pure sector-by-sector copies.

Leo: In fact, that's what we do on the TiVo before you mess with it. You use DD to do a dump of the data. I think that if you want to read more about this, there's good forums on this subject at DealDatabase.com. It's DealDatabase.com/ forum, and they talk about, mostly about TiVo, but they do talk about the Dish DVR. It's not as hackable, I know, as the TiVo. But I'm pretty sure you can make an image backup of it.

Steve: Yup. And in fact DealDatabase is the right place, Leo. I've done extensive hacking on my TiVos, as you know; and that is, like, where I go.

Leo: But nothing was as easy to hack as a Series 1. They've gotten so much - it's just they've messed it up, basically. Which was their intent.

William Dittmann in Northwest Indiana raises an important point: I just wanted to tap the brakes a little bit on your talk about VeriSign's PIP system. The VeriSign system you

describe bears an evolutionary resemblance to Passport. We mentioned this, that's the Microsoft single sign-on. While I'm sure VeriSign's been diligent in their system, it's still a federated ID system which has the side effect of severely impacting privacy and anonymity concerns. If you use this system, you're essentially making VeriSign a party to all your web memberships/transactions. In essence, everything is traceable back to me as an individual.

While this type of solution is very good for solving the problem of proving who I am, the problem for most people on the Internet is more like proving I'm the one that opened this account. And not related to the guy who opened the other account. I thin it's important to point out to your listeners that this is actually a concern with these kinds of systems. It's a privacy issue. And it concerns me because federated ID systems grow in popularity, and alternatives that may solve the other problem will not be developed. As I see it, federated ID systems are information brokers' dream come true. While I know I can create multiple IDs, there's little benefit because an information broker is more capable of managing multiple identities than most people are. He raises an excellent point which we didn't address, this issue of privacy.

Steve: Yes. And in fact I've got on my notes here in my outline for future episodes, coming up very quickly I'm going to do - we will really talk about this. I call it "The Dark Side of OpenID" because it's not just VeriSign. Even OpenID has this issue that is really worth, you know, mentioning and talking about, and that is that what we've been talking about, we're talking about a three-party system where you've got the end user, the site you're wanting to verify yourself to, and then a third party that provides this centralized credential verification service. I mean, the value of it is that you are able to authenticate yourself with a single site.

The problem is, in this scenario, as we'll remember as we talk about OpenID, you're authenticating with, well, you're going to a site you want to log on to. Then that site sends your browser to this third party, well, and to use William's word, ID federation site. And then it gets - so you verify yourself with that site. Then your browser is sent back to the site you want to authenticate to. And then that site checks in with the federation server to verify that this all went as your browser says it did.

The point is that that single point of contact is the value of using a system like this, but it's the vulnerability. Because just as we've talked about, for example, using third-party cookies and web ads, where you're able to be tracked on the Internet, essentially whatever server you use for OpenID, whether it's VeriSign or any other system. That server knows you because you've established an account and some sort of identity with it. And it knows every place you log onto using it because it has to have a connection as part of the protocol to the site you're logging onto.

Now, what's very cool is there are solutions to that which we'll be talking about in the future. There are ways for identity to be verified that don't use this, like, this lovers' triangle, essentially, where everybody has to know everybody else in this three-party system. There are ways to avoid that. But OpenID doesn't do it. So it's something to be conscious of.

Leo: So what are the other ways? Now you opened up a whole new...

Steve: Oh, it's very cool stuff, Leo.

Leo: Let's do another...

Steve: Oh, yeah, believe me, it's not something we can talk about now. There's a category of

techniques known as "zero knowledge proofs" where you're able to prove that you know something without giving away any of what it is you know.

Leo: I remember these going back a ways, actually, there were some systems to do that. And you know what, our correspondent may be right, William might be right that all the attention on these single sign-on solutions may have eclipsed interest in these zero knowledge solutions.

Steve: And as a matter of fact that was one of the things that he said that I did want to point out. I'm glad you brought us back to it. I think that was a very salient point he made, and that is that by this growth in popularity of something like OpenID and VeriSign's PIP solution, which it's here, it works, it's easy to get excited about it. It's very convenient, but recognize that there are some privacy aspects to it that you just need to be aware of. Again, I'm not saying that it's like, oh, no, this is evil and bad. It's like just, you know, recognize that that's what's going on.

Leo: Yeah, yeah. Boy, somebody's mad at you. He says you're extremely arrogant, Steve.

Steve: I know.

Leo: I don't even want to read this.

Steve: Yeah, it's okay.

Leo: You're very patient.

Steve: I chose it.

Leo: You did. This is somebody from North Carolina who does not give their name, and you'll see why: Steve, why don't you - I'm going to read it in his character. Why don't you put your eCommerce system up for sale? There are a lot of people out there who'd kill to have such a secure eCommerce system other than PayPal and Google Checkout. And not offering it up for sale is extremely arrogant. I believe that eCommerce is the future, but most eCommerce systems these days are exactly how you described them. Not everyone has a sophisticated eCommerce system, so they use the next best thing, PayPal. Heck, I'd tell my friend who runs a pet supply website to use your system and make things much easier on her because she had a really bad security issue with PayPal. Long story short, she got caught in a PayPal scam. If more websites used secure eCommerce systems like yours, it'd make things about 3,000 times easier on everybody. Please excuse my annoyance, but I can't stand it when something far better comes along, and the guy or gal who invented it says, oh, I'm not offering it for sale. There. I'm done. Signed, Anonymous.

Steve: Well, I really didn't mean to come across arrogantly. But it's the case that I have a model for GRC which is working. I really have an individual end-user orientation. That is, what I like to do is solve problems for the listeners of our podcast. And I've got nothing against pet supply stores. But...

Leo: Can I answer this for you?

Steve: The problem is that, I mean, and I've had other friends who have said, hey, Steve, why don't you just charge $2 for all of your freeware instead of giving it away for free? And the problem is there are, you know, money is not free. You pay a price for doing these things. And there's just - it's not practical to sell freeware for $2. It doesn't make any sense. Similarly, on sort of the other side of the scale, it's just not practical for me to sell an eCommerce system. We would probably sell maybe ten copies, yet it would take me a year to package it and support it and document it and add all the features that other people would want.

I wrote something that works just perfectly for me, for me, and not as a general purpose solution. So the fact is it wouldn't work for other people. For example, I imagine that pet supplies need to be shipped somewhere. But I don't do taxes on my system because software is a nontaxable transaction, and so I don't have to worry about dealing with collecting tax, worrying about what county people are in. In California, sales tax differs depending upon what street you live on. I mean, it's a real nightmare. So I wrote something specifically for GRC that works perfectly for us. And I wanted to share the problems I encountered and the solutions I came up with as part of this. But it would be such a different problem offering a solution for sale than it was just for our own purposes.

Leo: I'm no Objectivist. I'm no Randist. But this letter is a perfect example of what Ayn Rand talks about. Just because somebody is capable of writing a system like this doesn't mean he owes it to everybody else. It's not trivial to take Steve's - what you just said, which is it's not trivial to take your system and make it available. And it changes his whole life. Suddenly your business is selling an eCommerce system. And you have to support it. You have to keep it up to date. I mean, if you choose not to make that your business, I don't think anybody has any right to expect anything else of you. So I don't think it's arrogant in the least. I think in fact it's the opposite. It's arrogant for anybody to say, well, just because somebody has something or can do it, he has the right to give it to everybody else.

Steve: Or the obligation.

Leo: Or the obligation. Not right, yes, obligation. Again, I'm not an Ayn Rand nut. But that's an example of where Objectivism is absolutely right on. Write it yourself, folks. You want something better, do it yourself.

Steve: And in the last episode I told you how.

Leo: Yeah. It's not like Steve's hiding stuff. By the way, do recommend you go to GRC.com/securitynow and read the show notes for 109 because he's put some source code on there. And I just think that's just a work of art.

Steve: Oh, thank you.

Leo: I mean, it's really - one of my favorite books of all time is called "Programmers at Work." It's I'm sure out of print right now as Microsoft...

Steve: It's a great book. I have a copy.

Leo: Remember that?

Steve: Yup.

Leo: And one of the things, it talks to all these great, famous programmers, like Gates is in there, Charles Simonyi of Microsoft, just these great programmers.

Steve: Wozniak.

Leo: Wozniak. And it gives code samples. And they talk about their craft. And it's just - it's inspiring to read what these great artists do. And when I look at your code, it reminds me very much of that. It's just a work of art. Programming is wonderful. If nothing else I think in the last episode you might have inspired some people to turn to programming because it...

Steve: Oh, we actually did have some feedback from people who looked at the screenshot of a chunk of code from Bam Bam where I deal with the way I handle encrypting and decrypting the Payflow Pro password, which is what we use as our merchant services for credit card clearing, because I didn't want the password to be sitting in the registry where, if anything ever crawled into the server, it would be able to see that. So the user can - the actual administrator, you know, me, I put the password in the clear. The first time the eCommerce system starts up, if it sees that it's in the clear, it encrypts it and then puts it back encrypted, and then it gets it and decrypts it. So it's just a cool little bit of code. But we did have a lot of people, I mean, a surprising number who said, wow, I didn't realize that Assembly code could look that way. I'm going to go take a look at it again.

Leo: Yeah, yeah. It's fun writing in - I've written some stuff in Assembler. It's very satisfying. Because it's bare metal. You're right there with the machine.

George in Texas wonders about the security of TOR, The Onion Router. He says he saw an article on Ars Technica about a security researcher who used five TOR exit nodes to collect logon passwords from unencrypted traffic. What's the deal? I thought TOR was safe. I'm going to add something to this because we've now heard reports of at least one, I think two TOR administrators, one TOR administrator in Germany who was subpoenaed by the German authorities, they took his machine and all the information on that machine. Of course the authorities had no clue what to do with it. In fact it was worthless to them. But it does raise the issue of people can get these individual machines. Does that compromise TOR?

Steve: Well, and in fact the guy you're talking about in Germany, this was the second time he had been harassed by the government. And he said, I'm sorry, but I'm no longer going to be able to host an endpoint node on the TOR system because of the problems that it creates.

Leo: And by the way, the German police got nothing from this. They're at the wrong point.

Steve: Well, so, okay. So let's explain what's going on here. First of all, this rather

irresponsible security researcher, I mean, it's www.derangedsecurity.com. I consider him a little irresponsible, maybe a lot irresponsible...

Leo: That should tell you something, just the name.

Steve: Yeah, DEranged Security, because he was running - he runs a bunch of TOR nodes. And he specifically wrote a packet sniffer to find high-value email logon credentials, that is, username and passwords for, for example, governments and embassies and other high-value targets. He put these sniffers on five TOR servers, and they collected thousands of email name/logon password combinations. Then, because he didn't feel that anyone would take this seriously unless a lot of noise was made, he published these things that he had gathered publicly on the basis that, first, nobody would pay attention to this problem otherwise; and that, well, these were all in the clear anyway, traveling around the Internet, so what's the big deal?

Well, the big deal was the U.S. government immediately stomped on him and had his site taken down. All kinds of people were really upset. And essentially he published the email logons for a bunch of embassy, government embassy email accounts, and a whole bunch of problems resulted, as you can imagine. So...

Leo: But underlying that, did he point up a security issue?

Steve: No. What he did was - and this is valuable. The reason I wanted to put this question up, first of all, a lot of people picked up on it and wrote to us about this, so I wanted to address it directly. What TOR does is it anonymizes users of the Internet. It does not provide end-to-end security, meaning that, for example, SSL, Secure Sockets Layer we've talked about often, it provides end-to-end security, meaning that when I use - when a GRC customer buying SpinRite wants to buy SpinRite, the first thing that happens is an SSL connection is created, securing all of their traffic from their machine to the GRC server and back. So nothing can be seen. TOR doesn't provide encryption except between TOR server nodes. But on the final node, after your traffic is bounced around between TOR servers, where it is encrypted, and the TOR protocol makes it extremely hard to backtrack, once it finally is done bouncing around, playing ping-pong around TOR nodes, the final TOR node decrypts it the last time, essentially takes it out of the final encryption envelope, peels that layer off the onion, so to speak, and the traffic is then emitted or injected onto the Internet in the clear, that is, as plaintext.

So the mistake that these embassy people were making is they may have believed that all of their traffic was encrypted by using this system, when in fact all that was being done was they were being anonymized, meaning that potentially they could not be backtracked. On the other hand, their email had their client IP and server name, and everyone knew, anyone who looked at this would know where they were logging onto and what was going on. And all their email even was in the clear apparently, the actual content of their email. So...

Leo: So he just pointed out how insecure their system was, really.

Steve: Well, he pointed out that people were using TOR for the wrong reason. They were - and that...

Leo: Yeah. They were assuming they were secure.

Steve: They were assuming, exactly, that it was providing them absolutely security on the 'Net, when in fact all it was really doing, all it was meant to do is to provide anonymity services. Now, the reason the guy in Germany has gotten into trouble several times is that it was for child pornography that he was arrested. What happened was that government officials were tracking back the IP of somebody who apparently was pulling child pornography off of a child pornography site. Well, this child pornography viewing end user was using the TOR system to provide him or her with anonymity for this web surfing that they were doing. And what happened was the child porn IP was terminating on this TOR endpoint, where it then became encrypted and was then anonymized.

So somebody on the outside of the TOR system saw that this node was apparently making these child porn queries, when in fact they were being made on behalf of somebody using the TOR system for anonymity. And you can't blame them in this case for wanting to be anonymous. I mean, they understood clearly that this is what TOR was used for. So the authorities went to this endpoint and arrested this guy, unfortunately. He did nothing wrong except he was running a TOR node endpoint. And this in fact is the great danger of running TOR node endpoints. It's actually there's a double-edged sword here. If you run the endpoint, then people on the 'Net will believe that it's your machine which is making these queries of potentially bad stuff, when in fact your machine is making them on behalf of somebody who wants to be anonymous specifically because they want to do things which are in many cases and many countries and locations illegal to do.

The other side of this is that all of the traffic which is coming and going from that endpoint can be scrutinized by anyone running the endpoint. And in fact something that is useful is that this DEranged Security guy has a list of example exit nodes that can read your traffic. And so he says nodes named devilhacker and hackershaven; node hosted by an illegal hacker group; major nodes hosted anonymously dedicated to TOR by the same person or organization in Washington, D.C. - each of these are handling five to ten terabytes of data every month; a node hosted by Space Research Institute/Cosmonauts Training Center controlled by the Russian government.

Leo: Wow. The Russian government runs a TOR node.

Steve: Yeah. Nodes hosted on several government controlled academies in the U.S., Russia, and around Asia; nodes hosted by criminal identity stealers; nodes hosted by Ministry of Education in Taiwan, you know, run by China; node hosted by major stock exchange company and Fortune 500 financial company; nodes hosted anonymously on dedicated servers for TOR, costing the owner between $100 and $500 every month, meaning presumably they're getting some value in return; nodes hosted by the Chinese government officials; nodes in over 50 countries with unknown owners; and nodes handling over 10 terabytes of data every month. So the point is that, you know...

Leo: 10 terabytes.

Steve: Yeah. People are using TOR to do things anonymously. But you're using - when the traffic egresses from the TOR network, you don't know who owns the node that it's egressing from, nor do you know what purpose they're using the node for. So it's worth mentioning that, once your data egresses the TOR system, it is no longer encrypted. It has been anonymized as the TOR system provides. But depending upon what you're doing, you may still be giving your identity away. And people of unknown ambition and goal could be looking at it.

Leo: I think that's the bottom line, the really most important takeaway from this is that

TOR is for anonymity, not encryption.

Steve: Yes.

Leo: Dave Solon in Amish Country, Manheim, PA, has a tip for our listeners. He says: Thanks for mentioning the PayPal/eBay security key. As soon as you did I snapped one up at $5, a great deal. I've been using it in the past week with no problems. Today I couldn't get onto eBay. It kept asking for a key number, I kept pushing the button, kept entering the new numbers. He tried PayPal, same problem. PayPal, however, lets you enter two sequential key numbers, and then let me in. I almost disabled the key but decided to try something at eBay. I remember you saying you could enter your password, it then prompts you for the key number. I did that, it prompted me for my key number, I entered it, and voila, I was in. So I just wanted to mention this experience. I hope it may help others with the same problem. Thanks again for your great show. Love Security Now! and This Week in Tech. Can I pull a Dvorak and plug k12geek.com/blog? Yes, you may.

Steve: I checked it out, it's a nice little blog, although it's a slow blog, so if too many people who are listening go there. But it's for K12 interested stuff.

Leo: Neat. He must be a teacher.

Steve: Yeah. So relative to PayPal and eBay and the key, it sounds to me - and this is just decoding what he said - as though maybe eBay has some sort of problem with the password/key number concatenation. Remember that you are able to enter your username, and then in the password field you type your password and immediately add the current six digits showing on your token. By the way, Leo, I've decided to formally call these "tokens" now.

Leo: I like the name "token." That's good. They're not a fob...

Steve: Not fob...

Leo: They're not a dongle...

Steve: ...dongle and everything else. They are tokens. So what may happen is that there's some sort of a parsing problem on the way eBay's servers are handling this. So I did want to pass this tip on to our listeners, much as Dave had suggested. If you don't append the six characters to the end of your password, then you will be prompted for it in a next stage. And as Dave reports, that allowed him to get through and get logged onto eBay. So just a little tip. And maybe it's a function of his password. If his password had some digits on the end of it, I'm just thinking that now, that might be the reason that he had a problem, so...

Leo: Yeah, yeah. He might always have to do it that way.

Steve: It could confuse eBay.

Leo: I have never had any trouble with it concatenating. I don't use it on eBay, though, so I don't know.

Steve: Nor have I. It's worked for me every time I've used it.

Leo: Anish in Vestal, New York has a RAID question: RAID-1 is supposed to protect against catastrophic drive failure. In the event that a drive fails completely, the other drive is still available as an up-to-date copy. This is the Striped RAID as opposed to...

Steve: Actually that's mirroring.

Leo: Oh, RAID-1 is mirroring.

Steve: Yeah. RAID-0 is striped...

Leo: That's what I've got.

Steve: ...where you get twice the size.

Leo: What am I saying. Mirroring, of course. What happens if the initial drive failure is a slow, gradual process of an increasing number of unreadable sectors? Wouldn't these bad sectors also be copied over to the RAID-1 drive, making it unreadable, too? And what about running SpinRite? What if it corrected bad sectors on one drive and not the other? Could that render the RAID configuration useless?

Steve: Sort of an interesting question that I hadn't thought about or run into before. The good news is RAID-1, that is mirroring, will do the right thing. The idea is you've got a controller, the so-called RAID controller, which will read from both drives. Normally it's reading different data. If it's a smart controller, and you make an access, like for some block of sectors, the RAID controller may read the first half of the block from one drive and the second half of the block from the other, so that it's able to give twice, in theory, twice the reading performance. When you write data, it then writes the same data back to both drives.

But what this means is if in Anish's instance or example, one drive was failing to offer sectors that were being requested. The RAID controller simply asks for those sectors it could not get from the first drive. I mean, that's the whole point of having a so-called mirror. You have a mirror image of the second drive, and it will then pull those sectors from the second drive. And anything written will still be written together to both drives. So there is no way for unreadable sectors on one drive to contaminate those sectors on the other.

What we suggest people do when they want to run SpinRite on a RAID is to temporarily take the drives off of the RAID, that is, just remove them from the RAID controller, stick them on the motherboard, normal non-RAID controller, and run SpinRite on each drive individually. The problem is it doesn't make sense to run SpinRite on the RAID while it's still in a mirroring configuration because, for example, those unreadable sectors would be deliberately obscured by the RAID being a RAID, but it's doing what it wants to do, which is...

Leo: They'd always be readable.

Steve: ...the good data from the other drive, exactly. So in this case you really - you want to remove that redundancy and that error correction by design from being an intermediary between the drive and SpinRite. You want to give SpinRite direct access to the drive so it's able to fix things up and also show the drive where it's got problems so that the drive can relocate those bad sectors.

Leo: So if you corrected that on one of the two RAIDs and then put it back into the RAID array, it wouldn't screw up the RAID array somehow? It wouldn't....

Steve: Correct, because SpinRite will never change the data on the drive. So it will not break your mirror.

Leo: Huh. Interesting. And you've done this.

Steve: Oh, yeah, we've got users doing it all the time.

Leo: It's just that for some reason it seems counterintuitive to me, like somehow they're just going to - anything that changes that first drive is going to confuse the array.

Steve: Ah, because it's not changing the physical sectors.

Leo: But it might move a sector around; right?

Steve: Well, no. It would - it's sparing it out. It puts a new good sector in the same location. So it never does move the data to a different sector number. It changes the sector underneath the data.

Leo: Okay.

Steve: It's very, very cool the way it works.

Leo: All right. I guess that makes sense, yeah. Dale in Saginaw, Michigan, quite frustrated: When I went to log in to a major credit card bank today, it insisted I provide the answers to three personal questions. I get this all the time. Get this all the time. It drives me crazy, too. Is it not enough that sometimes I don't know the answers to any of the questions it offers? So today, when I finished giving and recording the answers to three of the questions and clicked Continue, I received this reply: "Please note: There was a technical problem in our system that prevented our storing the answers to your security questions. Since we were unable to record them today, we will ask you to select a new set of security questions at another time. You will not need to remember exactly what you had entered here. We sincerely apologize for the inconvenience." I'd change banks. That's it.

Has anyone ever considered that this whole process is just a continuing escalation? The more private information we give to all those databases, the harder it will be for us to verify information that someone else doesn't already know or have access to. Why don't I just give them my FOB number and be done with it? I don't know what an FOB number is.

Steve: I think he's talking about a fob.

Leo: Oh, my fob number.

Steve: Yeah, he's talking about...

Leo: Oh, but we've changed the terminology. It's now a key.

Steve: It's now a token.

Leo: So I've had this happen, this happens to me anytime the bank just feels like it, just decides, you know, for any apparent reason it'll ask me these questions again. And I never get them right.

Steve: You know, the first thing that occurred to me was, if I got something that said there was a technical problem with our system that prevented our storing the entries, blah blah blah, I would make sure I was not being phished.

Leo: Oh.

Steve: Because, I mean, certainly it's the case, I mean, it could be that something's wrong with their server. But...

Leo: We've seen this, you're right, yeah.

Steve: It could also be that somebody wanted to get information from you that they might be able to use in the future. But since they weren't actually talking to the real server, they weren't able to store that information on this credit card banking location. So, I mean, again, it's certainly the case possibly that there was a technical problem. But if you get something like this, as a Security Now! listener...

Leo: That's a red flag.

Steve: It is your duty as a Security Now! listener to run through the things you would tend to do to make sure you were not being phished, meaning you might try to go back to that prior page, make sure that you have an SSL connection, verify the certificate, and follow the chain of trust on the certificate back to somebody that makes sense to have signed that certificate, you know, not the Hong Kong Post Office - actually I'm kidding there.

Leo: That's probably safe.

Steve: That actually probably is. But I would, if I got something like that on a banking site, it would immediately raise a red flag, not just that their database was screwed up and they needed better programmers on their server, but just make sure a little phishing is not going on, and maybe in fact make sure that's really the site that you were on. But again, certainly I agree with Dale's point, and that is that this kind of nonsense is in fact spreading personal information around, and we know that it's not safe to tell everyone the name of your first pet or your favorite high school teacher or...

Leo: Although I should point out you don't have to give it the same answer every time. You just have to keep track of what you've told them.

Steve: Exactly.

Leo: Probably would be prudent not to. My favorite pet changes every time.

Steve: I have a lot of them.

Leo: But then you have to keep track of it some way.

Steve: Oh, and then it's worse than passwords.

Leo: It's worse than passwords. Are you ready, Steve Gibson, for the last question?

Steve: It's wacky number 13.

Leo: You know, we should do this every time.

Steve: I think so. We have wacky ones. I love the idea.

Leo: Well, we're going to get them now. This is from an anonymous listener in Arkansas because we don't want to name names: I've been thinking about this since your TrueCrypt episode. After I heard that episode I downloaded and installed TrueCrypt. We recommended it. It's a free encryption program for Windows. While going through the process of setting up an encrypted volume, TrueCrypt complained that my standard password wasn't long enough. Okay. I complied. I created a longer password I thought I could remember. And there's a big word, "thought."

Weeks later I went back and tried to mount the volume I'd created, and I wasn't able to remember the password. Now, fortunately there wasn't anything of value in that volume. But it got me thinking, how do I go about securing my digital documents with some kind of securely complex password that I wouldn't be able to forget? Also, what if I have head trauma and can't remember? Or worse yet, what if I die and my family needs access to my

documents? Which is a really important question. Here's what I came up with. Oh, boy. A tattoo.

Steve: He really wrote this, Leo. I'm not making this up.

Leo: Not just any tattoo, my friends, a blacklight tattoo. My idea is to take one of your generated passwords and have it tattooed on a rarely exposed part of my body with ultraviolet ink. This would - talk about a private key. This would allow me to always have my password with me, but it wouldn't be visible in normal light. I also thought it would be good to split the password up into eight eight-character chunks. All over his body. And then he could create passwords out of different chunks. And all he'd have to remember is, like, wrist, toe, ankle. And he'd have a good password.

Steve, what are you talking about? This is a brilliant idea. He says: I know it's not good for a spy or someone hiding from the government. But I'd like to hear what you think for your average Joe that wants to keep his tax return private. Love the show. Now, I'm sure it's a little tongue-in-cheek. But that's an interesting idea.

Steve: Okay.

Leo: Well, the problem is you can't change it...

Steve: Can you imagine, you go to the GRC Perfect Passwords page and get one of those 64-character nightmares, and then chop it up into eight eight-character chunks...

Leo: See, that's what's inspired to me.

Steve: And then maybe like an eight-by-eight block. And then you go to your local tattoo parlor...

Leo: Yeah.

Steve: ...and say, okay, do you have any UV ink?

Leo: Here's what I want. Now, you have to trust your tattoo guy.

Steve: Oh, you sure do.

[Talking simultaneously]

Leo: ...you go to eight different tattoo guys.

Steve: What kind of a lun- oh, yeah, good point, because you - no, but the problem is the

eighth tattoo guy, in order to tattoo you with UV ink, you need to do it under black light. So he'd be seeing...

Leo: Well, you keep your pants on.

Steve: Ah, that's - no. Now this is a reason - you're right, Leo - for putting them in different locations. So you say, okay, now, I want you to tattoo these eight characters on the bottom of my left foot. And the other guy does it on the bottom of my right foot. And in my left armpit - I guess you'd have to shave for this...

Leo: I'm not thinking it's such a bad idea, Steve Gibson, I might just do this.

Steve: Quite strange, Leo.

Leo: Better than getting a Nike swoosh tattooed on your hip.

Steve: Oh, god. And then when it comes to actually, you know, mount your TrueCrypt volume...

Leo: Oh, wait, excuse me, I have to take off my pants here.

Steve: Depending upon how secure your password is, you might have to completely disrobe in order to get access.

Leo: And find an ultraviolet light.

Steve: Yeah, that's a very good point.

Leo: Might be easier just to write this down.

Steve: Yes. Anyway, it's an interesting thought.

Leo: Actually the real problem is it's too permanent.

Steve: Well, exactly. It's very permanent. That's the other thing that I thought of, you know, very much like a retina scan or an iris print or fingerprint. It is something that's very permanent. Also there aren't that many permutations of eight by eight, you know, what is it, eight times seven times six times five times four times three times two is the number of possible sequences of eight chunks. So if you got knocked on the head or die, first of all you'd have to explain to your attorney, I guess, okay, now if anything happens to me...

Leo: Look under my - look at the corpse.

Steve: Take all my clothes off, shave my armpits, get a black light, and here's the sequence.

Leo: Oh, and you'd better hope you don't sag.

Steve: Right foot, left foot, right pit, left pit, and then...

Leo: I don't think so.

Steve: ...on from there. So anyway, I did want to share that because this actually was posted by a listener in Arkansas.

Leo: I love it. I think it's hysterical. And I'm sure he was tongue-in-cheek when he wrote it. But it's very funny.

Steve: It's definitely an interesting concept.

Leo: This is why we love our listeners and I look forward to these Q&As.

Steve: Why I couldn't stop reading them today, Leo.

Leo: I love them. Steve, it's so much fun. I do thank you so much for joining us, as always, and talking about security. And I congratulate you once again on making the very best, the very best tech podcast in the nation. In the world. Let's say in the world.

Steve: You know, I'm just annoyed that TWiT and MacBreak Weekly and Security Now!, basically a bunch of your podcasts are all in the same category, so we had to fight each other in order to get this. It's just, you know, I wish there were a different arrangement of podcasts. But again, I thank our listeners. They are the ones who made it happen. And hopefully they'll do it again next year.

Leo: Okay. They will. I'm sure they will. Yeah, it's great. Well, we kind of own the category, I would say.

Steve: And I'm going to say, Leo, among all of these emails that I'm reading, there are so many people who are saying please, please, please never stop. Never never ever never stop doing this weekly podcast. They are loving it.

Leo: So, life sentence.

Steve: Rest assured, listeners, we're enjoying it. So it's fun for us, too.

Leo: All right. Hey, Steve, a great pleasure. We'll do this again next week. Thanks for joining us, everybody. We'll see you next time on Security Now!.