SECURITY NOW!

Transcript of Episode #350

Q&A #142 / Cloud Security

Description: During this special Q&A episode, Steve and Iyaz host an entirely Twitter-driven Q&A episode, caused by the flurry of interest created by last week's focus upon Cloud Storage Solutions. After catching up with the week's security-related events, they zip through 21 tweets, then focus upon and examine the security architecture of one controversial and popular cloud storage provider: Backblaze.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-350.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-350-lq.mp3

SHOW TEASE: Coming up is Security Now!. Leo is out, but I'm here. But Steve, thankfully, will answer all your questions via Twitter. He'll cover the latest security news. And he's going to answer the question: Is Backblaze really secure? No, is it really, really secure? Steve knows, and he's going to tell you.

IYAZ AKHTAR: This is Security Now! with Steve Gibson, Episode 350, recorded April 25th, 2012: Q&A #142 / Cloud Storage.

It's time for Security Now!, the show that keeps you safe online. Now, obviously I'm not Leo Laporte. He's off in Norway. I hear he's hanging out with some fjords or something, taking some photos of them. If you're following him on path, you probably are seeing this, or on Twitter. But I'm Iyaz Akhtar. I'm filling in. But thankfully, of course, we have the star of this show. We have Steve Gibson, you know, from GRC and SpinRite, pretty much, has pretty much saved you so many hassles because, if you listen to him, and you watch this show like I do, you know that you get so much smarter just listening to Steve. So, Steve, thanks for being here, and hopefully you can guide me through this episode because I'm new to hosting Security Now!

Steve Gibson: Well, as Tom has demonstrated, there's really nothing to it. So I'm sure this will work very well. We have - our normal protocol is to alternate every other week with a Q&A. And this is nominally - and so this would be a Q&A week, following up on last week's topic, which was Cloud Storage. But the cloud storage topic in general is just so big and so hot right now. And of course just in the last day or two Google did finally unveil their long-anticipated Google Drive. Almost in synch with that, Microsoft upped their SkyDrive storage to 25GB free for anybody who'd been using it before as some sort of loyalty bonus or something.

And naturally security is a huge, I mean, maybe the huge question surrounding cloud storage because it's one thing for us to have our own local hard drives and be, like, trying to keep our machines free of viruses. It's another scale of security concern to be sending files that we may have varying degrees of concerns of privacy about out into the cloud.

So the good news is we absolutely know that we have the technology to do this safely. Crypto provides us absolute - the acronym that we've developed here on this podcast is TNO, Trust No One, meaning that we're able to send off blobs of noise, pseudorandom noise, which no force on Earth, as far as we know, can reasonably decrypt, and put that out there for storage, and then get it back. And so last week we talked about a number of these things.

Well, the topic and the issue of security generated a huge response on my Twitter feed. And so I began noting these. And I normally have a, like sort of a "From the Twitterverse" section of the podcast every week where I grab a few of the most significant, or tweets that I think would be interesting to our audience, and share them. Well, this time it just dominated the topic, the issue of cloud storage. And many people brought up very good points. So this Q&A is the first one we've ever had which was 100 percent driven by Twitter, rather than the normal Security Now! slash feedback, or I guess it's GRC.com/feedback, feedback page. I didn't even get to the mailbag.

But then something else happened, which is several people who are our listeners are using a cloud storage service called Backblaze. And it's not one that we covered last week of the dozen or so that we did. And when I went to - and so they tweeted, saying, hey, I've got more than, like, a terabyte. Like I think one guy's got 1.2TB up at Backblaze and is very happy with it. And I went to the site and dug around, and it did not look to me as if they had done their security right. I mean, not really wrong, but their pages were very confusing. Pretty pictures, but it just didn't look right to me.

And so I tweeted a few tweets that I'll read toward the end of the show which generated even more furor. And the CEO and cofounder of Backblaze sent me a tweet saying, hey, Steve, I'd like to understand what your concerns are. So because this is an important issue, I wrote him a note outlining what my problems were, and he responded with a detailed reply, and I responded.

So I want to catch up on the news, we've got some news this week, and run through these 21 tweets of comments pretty quickly. And then I want to look closely at Backblaze because they look like they're a great company. There's no reason to mistrust them. Their security is as good as many other cloud storage solutions. So, I mean, it's not like this is horrible and everyone should go running away from them. But they say that they're offering something that they're not. And that's the key. This is, unfortunately, this is a perfect example of people using powerful crypto tools, but not in the right way.

And so it's sort of a great little case study. Again, I'm not suggesting people run from them. I'm just suggesting that, unfortunately, their documentation is so poor that people believe they're getting security that they're not. And that often happens. I mean, I don't mean to be singling these guys out. But that was the question that I got asked by Twitter followers. And for everybody, I think what they're doing and the way they're doing it is just a perfect little case study. So that's what we're going to do this week.

IYAZ: So we're going to go through all these tweets first, or...

Steve: Well, let's catch up on news. There is, just in the last day or two, Microsoft Security Essentials came out with v4.0. Mary Jo Foley reported in ZDNet that what she had had was 2.something. And she wasn't sure whether they skipped over 3, or maybe she hadn't updated to 3 or what. But 4.0 is now available. It is showing as "important," an important update in the regular Windows Update deal. So you don't have to go get it. But if you Google just MSE, just the acronym, Microsoft Security Essentials, it's the first link that comes up.

And so you can manually download it, or you can go check with Microsoft Update or Windows Update, depending on what it's called on your machine. You'll find it listed, currently, not for immediate self-download and install, but you can check it, and then it'll download. It's about 12MB. I think it's 10 for the x86 version, 12 for the 64-bit version. And Microsoft says it's better, so that's what we should use. When I fired up this machine to set up our Skype connection, I verified that in fact it was available. I downloaded it. Install did not make me restart the machine, so that was nice.

So I'm just letting everyone know. And a tip of the hat to Simon Zerafa, who told me, gave me the heads-up that this thing was available. So it's, as we've discussed this before, Security Essentials is Microsoft's antispyware, antimalware, antivirus tool, sort of for people who refuse to purchase one of the alternative commercially available ones. This one is free. Some head-to-head lineups show that it's not quite as discriminating as other tools. It doesn't quite find as much as the commercial guys. But I say it's a lot better than having nothing.

IYAZ: Yeah. Microsoft's big thing, I mean, Windows is so popular, it gets attacked all the time. And when Microsoft went ahead and put in their defense stuff, I mean, that's important. You really have to have [indiscernible]. And having something is better than nothing. So even though people knock it, just have something. I mean, that's always safe.

Steve: Yeah. And I've got a good friend who someone else set up a Vista machine for her a couple years ago. And there was a trial version of I don't know what, Trend Micro or something. And it worked for however long the trial is, 90 days, six months, a year, who knows. And then it began bothering her for money. And so that's when I got the call saying, "Steve, this thing is telling me that my trial of Trend Micro has expired. Do I have to pay them money?" And my answer was, eh, no. Remove that and install Microsoft Security Essentials, and you'll never be bothered for money again. So for someone like that, who's not a hardcore security person, who otherwise would have nothing, this makes more sense, especially if paying something for it is more than they're willing to do. So, yeah, I completely agree, Iyaz.

Also Firefox v12 has happened. When I - I was at 11.0, and I have famously recently made the move from - I was way back. I was staying back in 3 land until they finally fixed their memory problems, which they finally did, despite several claims to have done so but not, until they got to 11. All I did was do - I went under Help and About. That brought up the dialogue and immediately prompted it to start downloading No. 12. So it's about a 7MB update. There's a number of ongoing fixes, bleeding-edge features and protocols that they're adding. They're heading toward what Google has done, which is the seamless, in-the-background update.

What's significant about 12 is that they have an architecture now using a service, a Windows service. The service is not running all the time, but they're able to start it up when they need to. It has the privileges of the system to allow it to update the files which you would normally have to pass through the User Account Control window to allow. So this is their way, with v12, they've bypassed User Account Control.

Now, I get - I have a little bit of what we call on the podcast a "Gibsonian reaction" to that because one hopes that they've done this safely and securely because potentially this means that there is a service that Firefox users now have in their system which, if abused, could install anything on their system. So I'm sure the Mozilla people know that and that they've - somehow they've created authentication and crypto and whatever they need to in order to do this securely. It certainly is a convenience that Firefox is able to do this. I did have to restart Firefox to go from 11 to 12. As I understand it, their goal for the next major release, v13, is they'll get full seamless background updates and just be fixing Firefox for us, very much the way Chrome does, where it's just always, as you're using it, you're using the latest version.

IYAZ: Steve, now, you installed Firefox 12. Is there an option to turn off the automatic updates after you turn it on? Because, I mean, if you have extensions and things, couldn't these things break over time because you're just automatically updating? I mean, I was cleaning out one of my old computers, and I found Firefox 1.5. And, like, I'm not letting it go. I might need this later on, just in case. But, I mean, can you turn off some of these, I guess, convenient features that could be a problem?

Steve: I'm sure it's configurable. When I brought up the Help > About box, it immediately started downloading it, although I did have to tell it to go ahead and do its thing and perform the update. It then did run through all my add-ons and verify that they were compatible with this version. So in this case, for me, going from 11 to 12 was painless. But I'm sure - I remember when we were talking about this initially, that they were going to be doing this, that this background update, because some people would be uncomfortable with it, or maybe corporations are still saying we need - we're a Firefox house, but we still don't want this happening without IT first taking a good hard look at what's going on. So that may very well have happened. That is to say, I'm sure there is a way that you can disable that behavior.

IYAZ: [Indiscernible] the automatic updates are just on the Windows version. As far as I know, the Mac version doesn't allow that just yet.

Steve: Ah, okay. Good to know that. Now, a couple weeks ago we were talking about iOS password managers in some depth. And one of the things that I ran across indicated that, depending upon your upgrade path from iOS v3 to 4 and to 5, it might be that you did not have the whole-device encryption actually running that iOS 4 and 5 have offered, due to the fact that it might not have been present, well, it wasn't present in 3, but it might not have gotten activated. And somebody was kind enough to send me a tweet with the details. Unfortunately, that got lost in my Twitter feed, so I can't give him credit. But on TidBITS.com is an article where the author explains exactly how this can happen.

Now, the good news is it's easy - both things are easy. It's easy to tell what your current state is, and it's easy to fix it if it's wrong right now. So on the Passcode Lock settings screen of any iOS device, phone or pad, if you see the words at the bottom, "Data protection is enabled," then you're golden. That means that you have whole-device encryption running, not just some parts of the OS being encrypted as Apple was sort of moving forward in how much of the device storage they were encrypting. So the Passcode Lock settings screen, if down at the bottom says "Data protection is enabled," you're good.

If you go there, and you don't see that, then you don't have whole-device encryption. The way that can happen is, if you had the passcode enabled at iOS 3, and then simply upgraded to 4, encryption would not have been applied. So there are probably some technical reasons that Apple was unable to do that, as evidenced by the fact that, in order to get it on, if it's not, you have to first - and so here's the steps:

Assuming that you're now on probably iOS 5, you first disable and remove the passcode from the device, so shut that down first. Then backup the device to iTunes. And you can just Ctrl-Click on it and then choose Backup in iTunes. So that'll copy all of the device onto your local iTunes-based drive. Then restore the device by clicking the Restore button on the summary screens in iTunes. And this article says that he's been told that Ctrl-Clicking and choosing Restore does not work. So don't use the context menu for restoring. Use the Restore button on the summary screens in iTunes. And once you've done that, then on the device reenable your passcode, and you'll then be secure and fully encrypted and of course go back to the Passcode Lock settings screen - that's where you would have been anyway to reenable your passcode - and verify that it now says "Data protection is enabled," and you've got whole-drive encryption running.

In this article he mentioned that he was the presenter in some sort of a conference of security professionals, and he had everybody in the audience look, and a disturbingly high percentage did not have data protection enabled, although they had passcodes on their machines that were running iOS 4 or 5. So that was a small sample, but these were security-aware people, and this had just caught people off guard. So I would imagine some percentage of our listeners are going to be in that condition, too. Now we've got the whole readout on that.

IYAZ: So there's no way to automatically do this. You've got to go through this five-step process. Which the first step, again, is to turn off your passcode. So it seems like, so, wait, to get more secure you have to be less secure for just a little bit. So there's no automatic way to do this. It's just simply you've got to do these steps. Just double-check, and then your whole device will be encrypted.

Steve: Right.

IYAZ: Okay.

Steve: Right.

IYAZ: So now I've got to check on my old devices. This is good. This is good.

Steve: Yeah, I think a lot of people are going to be doing that. The good news is it's easy to see. If it had been tough to determine whether you were encrypted or not, or if you had to do this, like, preemptively, and not know one way or the other, that would have been a big pain. But it's easy to see whether you've got it already encrypted or not, and not that big a problem to fix it. I imagine everybody who is a podcast listener, like this podcast, cares. And so this will let them fix it easily.

Also, on the Mozilla front, something that we have talked about and anticipated has happened. I got a tweet from Ron Chung, who his tweet is @directorronc, that Mozilla had released - in fact, I think when I first saw it, it was at beta, and now it's released - their native PDF viewer. As we know, Chrome, one of the nice things about Chrome is they've got a PDF viewer built in. You're not using Adobe's rather rickety platform for viewing documents.

So it was good news that Firefox would be getting a native PDF viewer, too. It does exist. They're calling it PDF - the file is PDF.js. And what it is, is a JavaScript and HTML5 - I consider it an amazing capability demonstration. The fact that you can render PDF files with JavaScript and HTML5 is phenomenal. The bad news is, it's still not very usable, and it's not clear to me it's going to be. If you absolutely refuse to use Adobe's plug-in, the Adobe Reader plug-in, and in fact I had the Acrobat plug-in that was a nightmare because it was, like, bringing up dialogues and complaining and all kinds of things. I was finally - I finally moved over to Sumatra PDF browser plug-in, which I really like. But for the sake of the podcast, I thought, okay, well, and I was hoping that this native HTML, I'm sorry, PDF viewer for Mozilla would be a good thing.

So first time I tried it, it didn't appear to work. Well, that was NoScript blocking me. You have to - and I should have looked, but I didn't. I just thought, okay, well, it doesn't work at all. Then actually I sent a tweet back to Ron, and he said, well, it did for me. So I poked it a little harder. And you have to permit something called "moz-filedata:" under NoScript, and I gave it permanent rights, and then it does work. But, ooh, is it slow. And again, it's amazing that they can do it at all.

So I would consider it a phenomenal capability demonstration, but there's no way I could use it because, as you scroll down pages, you get blank pages with a little wait spinner clock, and then the page renders. And it does a very nice job, once it gets there. But it's just - I don't know that it's ever going to be practical to render something that is written in a language as complex as, well, it's Postscript is what underlies the PDF format. So this is JavaScript interpreting - which is largely interpreted - interpreting Postscript in order to do very sophisticated rendering.

To me, it's - you're always - a native solution, like Sumatra or like Adobe, is just going to do a far faster and superior job. And given that we have Sumatra browser plug-in, which is free, and if nothing else it doesn't have as big a target painted on it as Adobe does with PDF vulnerabilities, I'm very happy with Sumatra. But they may be able to advance it in the future, make it go faster, who knows. Or maybe they'll end up writing a native PDF viewer themselves, much as Google did with Chrome. But it does exist, and it's cool that it can even be done.

IYAZ: Any alternative to Acrobat is always pleasant. I mean, I remember when I'd see links to PDFs, it used to say, "Warning: PDF" because that means it's going to try to launch an application. And you might see that splash screen that says it's loading, like, a thousand plug-ins, and you're like, why does it take this long? When Chrome came along with its built-in reader, it was actually pretty easy, to the point where there was a little JavaScript bookmarklet that you could say open with Chrome's or Google's PDF reader. And now that Mozilla's got it, I mean, it's going to make it a lot easier to accept PDFs instead of going, I don't want to open that. That's just completely...

Steve: Yeah. Well, and of course, as we know, Adobe was trying to promote all their formats. So they've got Flash built into their PDF reader. It's like, okay, wait, wait, wait. When did a document ever need to have Flash in it? And yet it's been a multi-opportunity vulnerability for malware and malware authors in the past. It's just a bad place. So of course the - well, and scripting, too, scripting in the PDF. So best practice is disable Flash, disable scripting in your PDF viewer, of all things. So it just makes much more sense to have a dumber PDF viewer because dumber in this case is going to be safer since it is a large attack surface for people who are messing around on the web.

So this was just tongue-in-cheek. Actually I tweeted something about "Here's a hoot" yesterday. And I learned that there are some script bots watching Twitter feeds. And if you ever use the word "hoot" in a tweet, then these things pick them up and retweet them to everyone in the world. So it's like, okay, well, it's the first time I had used that word.

Anyway, Sophos, the well-known and good reputation security firm, put out a press release just recently that announced that - and this is what I thought was so bizarre - one in every five Mac machines harbors Windows malware. And they have a link where they show a breakdown. They took, using their own assessment tool, which was resident in 100,000 Mac machines, so they had a 100,000 Mac sample, one out of five of those machines contained Windows malware. And so it's not going to be effective on a Mac. It's like the wrong DNA. It can't infect a Mac because all of its tricks are for Windows.

But of course what this represents is that there's just so much of this Windows malware around, and that something that's trying to infect you, for example, using a Java exploit, which of course Windows has shared - I'm sorry, Mac and the Apple have until very recently shared with the rest of the Java-using, largely Windows community. When your browser has that vulnerability, the thing trying to infect you either doesn't know or doesn't care who the host is. Chances are still, due to the install base of Windows versus Macs, that it's going to be a Windows machine, all other things being equal.

So it's just sort of bizarre that 20 percent of Macs have Windows malware on them, just because it's out there, and they picked it up surfing and doing things that were unsafe or with some sorts of add-ons, like maybe a Java exploit that allowed the stuff to get in. Once it got there, it looked around and said, oh, crap, I can't do any - can't get up to any mischief. I can't install myself as a rootkit or do whatever I was going to do. But it's still a file sitting there.

And after that tweet I got two responses: Rene van Belzen, who's in Bergen op Zoom, Netherlands, he sent back: "About those Macs with Windows malware being ineffective, they still form dormant, yet potent repositories for Windows malware." And of course he has a point. I mean, it is bad, and you don't want it jumping off onto Windows machines on the same network or through any sort of vector. So even though it's unable to get your Mac machine, it's still sort of latent evilness that's sitting there on your machine. So if you find it, by all means, remove it.

And then William Ricker in Boston, he said: "Unscanned Macs in mixed environments serve as carriers for Win malware if they have email or USB." Which is sort of a variant on what Rene said. So certainly good points. Even though it is the wrong machine that's infected, it's still not something that you want around.

IYAZ: Yeah. When I first heard this story, the first word that came to mind was "carrier." It's just like, okay, it's got this little disease. And people who have Windows already were a little bit not so happy with usually OS X counterparts. But now they have even more reason to be very suspicious when they're like, where's that USB key been? What files are on this thing? Because you try to open everything, you might find out that it's infected. I mean, it's almost because OS X users have been lulled with the security, like, hey, look, nobody goes after us. But they're downloading files like crazy from who knows where. They can do sillier practices. But they are downloading some infected files that, if passed around, could lead to disaster.

Steve: Yeah. Another little tidbit coming back to something that we discussed years ago. It came to light that the Firewire interface, the IEEE 1394 interface, which is pretty much fading into the background now that we have USB II and especially USB III, and what is that, is it Thunderbolt or something, whatever Intel's insanely fast next-generation serial interface is.

Anyway, what we learned, and we have discussed in the past, is that the Firewire interface is actually a serial bus connection to the machine it's installed on, and you have things like Direct Memory Access, so-called DMA. And again, Simon Zerafa brought this to my notice, and so I want to thank him for it. The abuse of Firewire is pretty much now going mainstream. Over at BreaknEnter.org, under their projects directory is something called Inception. And quoting from them, from their page, they said: "Inception is a Firewire physical memory manipulation and hacking tool exploiting IEEE 1394 SBP-2 DMA." SBP is just the acronym for Serial Bus Protocol, which is all Firewire.

Then going on it says: "Inception was originally coded as a GPL replacement for winlockpwn, the Windows Firewire unlock tool made available by Adam Bolieu, a.k.a. Metlstorm. Winlockpwn was quite stable against older Windows XP targets, but did not perform well against more modern operating systems like Windows 7, and it is not maintained anymore. As of Linux kernel 2.6.22, Linux Distros ships with the new Juju Firewire stack, making winlockpwn obsolete. Thus Inception was born.

"Inception aims to provide a stable and easy way of performing intrusive and nonintrusive memory hacks on live computers using Firewire. It is primarily intended to do its magic against computers that utilize full disk encryption such as BitLocker, FileVault, TrueCrypt, or Pointsec. There are plenty of other and better ways to hack a machine that doesn't pack encryption."

So to briefly refresh, the idea is that one of the forensics tricks that law enforcement will use or bad guys can use is to gain access to your machine while BitLocker is running, while TrueCrypt is running, while FileVault is running. The point being that the cryptographic keys are available, they have to be available in real-time in order to be shuttling encrypted and decrypted data back and forth between your system and the hard drive. And there is now rather well-developed technology that knows immediately where to look. And if you give something access to your RAM, these keys are in RAM in the clear and can be immediately snarfed.

So the takeaway from all of this is - and we've said it before. This reminded me to say it again. Since Firewire is probably no longer an interface that you're using, if that's the case, just remove its drivers. Maybe in the BIOS, if you can turn off the IEEE 1394 interface, disable it there. Or remove the drivers, or disable it in the OS. You just don't want it on. It is a port into your system's RAM. And standard security best practice, of course, is to disable and turn off things you aren't using and don't need. This is near the top of the list for machines that have a Firewire port. I mean, again, it's not like it's huge danger. The hackers can't reach over from the Ukraine and access your machine's Firewire port. But for physical access to the machine, if you really do have sensitive stuff there, this gets right in.

IYAZ: So for users who don't have a Firewire port, should they be worried that somebody can just put in a PCIE card and put Firewire into their computer and then be able to access stuff? Or is that like a crazy scenario?

Steve: Well, no. In fact, I don't think that is such a crazy scenario. Some of the Macs, for example, have the Firewire adapters for the PCIE cards that you're able to slip in. And that probably gives you the same sort of access. That is on the system bus. So best to just disable those things, if you don't need them.

IYAZ: Start putting hot glue into all the ports and PCIE slots. You're like, forget it, nobody's getting in here. Physical access always is trouble.

Steve: Yeah, it is. I mean, and we've talked, for example, in the past about that whole freezing the RAM and taking it out of the laptop and then putting it into a different machine, like keeping it alive by bringing the temperature down. It's like, okay, well, there's a physical access hack if ever there was one.

Once again I have to punt on that Dropbox tech blog entry where they talk about the realistic password strength estimator. As you'll see, when you see what's in this podcast, I just did not have time to get to it and had no time during the week. So I'm not forgetting it. I'm pushing it into next week. And I will keep it alive until I finally get to it.

Okay. So, quickly, 21 Tweets from the Twitterverse. Ahto Jussila, he says: "Tarsnap is the only cloud tool I feel comfortable installing on server. Client source is available. There are bug bounties and a well-known author." And so my reaction is that's 100 percent understandable. Tarsnap is - it's Linux-y or UNIX-y. So, I mean, and it's only for those platforms. And I agree. I like the idea of the visibility of an open source tool. And so I just sort of wanted to acknowledge that posting. Tarsnap was down near the end of the list because we did it alphabetically last week. And I just sort of shrugged it off as, well, this isn't really for anybody.

But I stand corrected. I think it is for UNIX and Linux admin-scale people who like the idea of a clean and simple and sort of knowable tool, rather than just assuming that they're linking this to something in the cloud, and it's going to be good. It uses Amazon S3 and EC2 as its backend. And of course I'm a big Amazon S3 user. They do double the price, which puts it at - Amazon is about $0.15/GB/month, and actually Amazon seems to have dropped their price. I think it's now $0.125/GB/month. Tarsnap doesn't give you an interface to your own Amazon account, but it charges you for your use. So they charge $0.30/GB/month, and they front for Amazon. So they're using Amazon themselves, but you don't have direct access to it, for example, as you do if you were using Jungle Disk or one of the other S3 frontends.

And they also charge $0.30/GB/month for transit, for bandwidth, apparently in both directions. Now, what's significant there is that Amazon and Rack- that Jungle Disk and Rackspace - no, I do mean Amazon and Rackspace as cloud providers are now both providing free upload transit. Last week I thought that only Rackspace was. But I checked, and Amazon has since dropped their upload fees. Now, to me that's significant because, for example, one of the modes I operate in, and I'm thinking of this from the standpoint of a server, is in the wee hours of the morning I have something that wakes up on my server and takes a static image snapshot of my entire GRC server hard drive. It's actually surprisingly small. It compresses it. Then it's encrypted, and it's sent up to Amazon. So the idea is it's a win for me that I'm not seeing any upload fees. I maintain a number of past images from prior days, and I have sort of a staggered backup policy that is implemented, so I've got a bunch of them there.

But so the point is that I am, every day, I'm sending a blob up. I'm never bringing it back. So all I'm being charged for is the storage, which is from my standpoint quite affordable. And Amazon is, like, rock solid in terms of availability of the data. They can have, the way their redundancy is set up, they can have two entire of their datacenters go down, and you still have real-time access to everything. So I'm quite bullish on Amazon. They're just so big that you get the advantage of that. And if you do your encryption on your end, then - believe me, there's no way I would be sending this stuff anywhere unless I was encrypting it first, and I was the only one who ever had the keys. But the idea of free upload really works in this sort of scenario where you're highly weighted in the direction of sending stuff to the cloud, only needing to get it on the occasion that you're needing to recover from some calamity or for some reason to get a backed copy.

Moving on, Greg Bell, who tweets as @ferrix, he said - oh, also about Tarsnap. He said, oh, yeah, he actually - I'm confused now. Oh, sorry. He tweeted about Tarsnap. He said: "Only charges transfer and storage, no monthly or yearly fee." Okay, well, that's a little confusing because they're charging for storage on a monthly/yearly fee. But I guess so he's saying that there's, oh, yeah, he's saying that there's no other charge like a static charge just for the privilege of their service. And he says: "I think that accounts for the price. Great for my UNIX backups."

And then the reason I was confused is Colin Percival, who saw that Greg had commented in my feed, so he wrote to both of us, and he said: "To the point where the median Tarsnap customer is only paying about $0.10 per month." Well, okay, I don't know how Colin has that data. Maybe he has some inside information about the amount of storage that Tarsnap users are storing. But that would be, given $0.30/GB/month, someone paying $0.10 per month would have a third of a GB backed up. So I suppose it's certainly possible to only be paying $0.10 a month and nothing more.

Carl Wilson said: "FYI, SpiderOak's synch requires data to be backed up online already." Then he said: "Sharerooms can be password protected. SugarSync has no PIE," meaning pre-Internet encryption. And so I'm not sure what he means when he says SpiderOak's synch requires data to be backed up online already. I'm going to find out because one of the things that has happened as, again, a result of my continuing focus on this whole topic of cloud storage, is that SpiderOak remains a contender for a very nice-looking, very feature-rich cloud storage solution. So I'm still - maybe next week. I just - it's a function of what time permits and how much time I have to dig down into it. But the SpiderOak guys are there. I've heard from somebody else since at SpiderOak who's standing by to give me any technical details that I need. So it's certainly something that I'm going to get to as soon as I can.

Faiz Imam said: "I have 1.2TB backed up on Backblaze for $5.00 a month, mostly photos. Took me months to get it up there, but it rocks." And so I think that this was one of the tweets that sort of began my saying, okay, what is Backblaze? And that drove me into taking a close look at them. And at the time that I made this note, I said, "So far as I can tell, it's not TNO." And that's 100 percent confirmed now. It's not. Which is not to say that's a huge problem. I don't want to overblow this. And lots of other providers aren't. For example, none of the big ones are. SkyDrive and Google Drive and Dropbox, of course, these big ones that offer lots of services, the notion of the services they offer are incompatible with them having no visibility into your data. You can't have it both ways.

So that's something that we'll cover later on in the podcast because I think what we're going to be seeing sort of evolving out of all of this is maybe sort of a hybrid solution where the user has control of some of the things which are absolutely black out on the cloud. Nobody can possibly get to them. And then other things, which they know are less secure, but that's the tradeoff they make because they want to be able to share links to them with their friends, or they want simultaneous document editing and all these other things that we're going to be seeing, cloud-based services in the future. Obviously, for those things, it just can't be an opaque block that the provider has no visibility into.

Thilo in Switzerland says: "If you review SpiderOak for Security Now!, you should also check Wuala, as they also do pre-Internet encryption with great platform support." So we talked briefly about Wuala last week. Leo had once been using it and then dropped it. At one point they were apparently putting their, essentially, their cloud storage was other people's hard drives. And I don't know definitively whether they are still doing that or not. What I think Leo said was that, if you gave them a chunk of your hard drive voluntarily, then you got better terms with them. They definitely have a technology which uses an ECC-like redundancy, which I did briefly mention last week, the idea being that, in very much like a RAID, where you can have a RAID drive die, and you can still reconstruct the data from the remaining drives, this is similar to that. And there are other technologies like that where they build in redundancy into this distributed spray of data so that, as long as there's some critical mass of that still available, they're able to reconstruct the whole.

The problem is, for me, it's written in Java. Now, whether it puts our data onto other people's machines, I just sort of don't like that. Again, that's a tradeoff a user could make. If it gives you a much better price, then maybe that's a tradeoff you want to make. But Java is - I just don't understand that. I could see Java things being useful, like free toys and so forth. But these people are charging for this service, and Java, as we'll encounter a little bit later, has some problems. Several of the providers are Java-based, of these cloud providers are Java-based.

And my point is, why, if you're making money on this service, then why not just do a native client? Write a client for Windows. Write a client for the Mac. And then you're not having to deal with the whole Java-esque problem. And part of this is sort of the nature of where this came from. It was a project of the Swiss, a security group in Switzerland. And it just sort of evolved into a commercial thing. Anyway, I don't think I can get around the fact that it's written in Java.

Oh, and then he said in a second tweet: Don't catch why you trashed Wuala in your latest Security Now! podcast. They don't do space sharing and seem to be on a par with SpiderOak. So it does look like they've done security right. But my comment to that was, yeah, but Java? It's like, rather not have that. And there's alternatives that don't.

Josh Mandell said: "I don't think I heard clearly if CrashPlan has pre-Internet encryption. But their FAQ states how they do it and even has a link to Security Now! #230 on Blowfish." Which I kind of got a kick out of, the CrashPlan people linked to this podcast where we described how Blowfish works because they do. And as I remember, they give you one level of Blowfish, like 128-bit key Blowfish for free. And then if you switch up to their paid version, they give you 4048-bit keys, which is the largest key size that Bruce Schneier's Blowfish cipher is able to handle. But it's also written in Java.

And then I quote, "How to use CrashPlan but keep" - oh. This was on a site. This is what I knew I had here somewhere. This was on a site that noticed the amazing, actually "ridiculous" is the word he used, amounts of memory. He says, "How to use CrashPlan but keep the Java process from constantly using ridiculous amounts of RAM." And he says, "I use CrashPlan on my online backup, and it's great - reasonably priced, secure, easy to use. But there's one problem. For better or worse, CrashPlan is a Java app. And one of the problems with Java apps is they are greedy pigs when it comes to RAM usage. I've often noticed that Java was using close to 400MB of RAM, even when nothing is happening. To be precise, when CrashPlan is just hanging out and not actually backing anything up."

So once again, for me, that's a deal killer. If I'm going to have something in my machine which is going to essentially be part of my machine, running all the time in the background, keeping things synchronized, I'm not willing to give it just shy of half a GB of RAM for that, not when native clients can be written and have been written that are way more lean.

Mike Dennis tweets: "I have a four-year family unlimited plan with CrashPlan and been pretty happy with it. I have over 1TB backed up to them today across my PCs." And so I wanted to balance the prior one with this. Mike is happy that he's got his four-year family plan and tons of data up there which is affordable.

I got a tweet from someone who didn't give me his name, but his handle is @Engrpiman. He's in Boise, Idaho. He said: "Jungle Disk web interface allows for Trust No One. You must enter your username and password. Then it asks you for your second, which is the optional password, and then you have files." And so my reaction is, well, I have not looked at Jungle Disk. I will probably send off a note to Dave, who's known as Jungle Dave in the forums, and ask him. The question is, where Jungle Disk is configured for TNO operation, you give it an optional password which then performs all local encryption of blocks that are sent off to either S3 or Rackspace. So that means that the data that they are storing, as is required for TNO, they have no visibility into. They have no keys, never get them, never will. All they can do is give you back the same blob that you gave them.

So the question is, in order to do a web-based interface, they would need to do what LastPass does for us in their implementation, which is that last decryption phase would have to happen in the browser, that is, client-side, not server-side. From just that description, you type these things into the browser. Well, off they go. Who knows where they go? Dave may say, well, of course, Steve, I did it the right way. You enter your password into the browser. You're entering it into a local JavaScript app. All of the decryption is performed locally. Nothing unencrypted ever goes over the wire, and that password never leaves your machine. I imagine that's what he will say, but I don't know that.

So we can't assume that this is TNO until we understand the technology, I mean, the details of the technology. And this actually segues perfectly into where we'll be headed later in this podcast when we talk about Backblaze and the way they use their crypto because the devil is in the details.

Thomas Scrace in London said: "Have you looked at Arq" - it's HaystackSoftware, all run together, HaystackSoftware.com/arq - "for backup? One-time purchase, and it backs up to S3." So I had not looked at it. I like it. If you're a Mac person, it's Mac only, but it looks very nice. It looks very similar to Jungle Disk. It does local AES-256-bit encryption. It looks like it's 100 percent TNO. It uses S3 as the backend, which, as you've just heard me saying, I like a lot, meaning you can send stuff up there till the cows come home and not get charged for it except the amount of static storage there. They have an open documented file format, which is - and an open source command line arq_restore command utility which is hosted over on GitHub. So that's there and in the clear. You can see how that works.

They have a one-time license of $29. And I should say this I think is the model that we're headed for, one-time license for this kind of stuff. I mean, there are going to be some very good solutions that you just pay some amount of money once for, and then - of course that's in the model, then, where you use somebody else's storage, like Amazon S3. So you are paying monthly for the actual storage and maybe the transit of the data. But it just feels wrong to me that we're stacking up all of these monthly charges on top of each other, somebody charging for the privilege to use somebody else's storage that you're also being charged for the privilege to use. Because we don't have to pay all that. Something like this looks very nice.

So again, this is HaystackSoftware.com/arq. They also do versioning. Under Versioning they called it a "Wayback machine." They said: "Arq keeps multiple versions of your files, a backup history. Following the initial backup, Arq automatically makes incremental backups every hour, every day, uploading just the files that have changed since your last backup. Arq keeps hourly backups for the past 24 hours, daily backups for the past month, and weekly backups for everything older than a month."

Under Features they wrote: "No limits. Arq backs up everything you tell it to back up." And here he's clearly responding to limits of the other services, limitations. For example, he says, under "Arq backs up everything you tell it to back up," it doesn't skip videos or ignore certain file types. It backs up files of any size - 4GB, 40GB, it doesn't matter. It backs up your external drives and your network drives. Several of the other services won't do either of those. It doesn't delete backups of your external drives just because you haven't plugged them in lately. And there are a couple services that actually do that.

It doesn't forcibly delete backups older than four weeks.

And then my note is, I noted it will even throttle upstream transfer rates, which means that famous buffer bloat doesn't bite you. While it's working, it will not saturate your outgoing bandwidth. So, as we know from our buffer bloat episodes, it won't screw up your downstream interactive use of your network connection while this is happening.

Anyway, it looks very nice. I'm really - I thank Thomas for bringing it to my attention. And for Mac people, check it out. It's certainly, if you like the idea, as I do, of where it makes sense, sort of more of a DIY sort of scenario, where you've got a relationship with a storage provider, and you kind of want to manage that yourself. You're firm about wanting TNO. You're sure you're not going to lose your key because, of course, if you do, you have no access to that. But that's the whole point of trusting no one. You have to trust yourself. And it's not a monthly plan that you've got to get onboard with. Anyway, for a certain class of our listeners, I'll bet this is a great solution. So check it out.

Brandon Furtwangler says: "Wow, SkyDrive just gave me 25GB free as a loyalty offer. Can you find out more about the security model? Could be my new fav." Well, I looked. And I looked, and I looked, and I looked. There's absolutely nothing available - this of course is Microsoft's SkyDrive - nothing available that I could find about their technology. No sign of any Trust No One. And as I mentioned before, it's highly unlikely, given the services that they're offering. The features that they're offering are far beyond here's a blob of opaque noise. Keep it for me. And if I ever need it, then I'm glad to have it back. And everything of vital importance is in there, so I can't risk anybody else getting it. So that's very different than these highly service-oriented cloud services, which is really where Microsoft is with SkyDrive and Google is with Google Drive.

We're going to encounter, and I think we'll be seeing a lot of the ideas of layering security on top of these services and creating sort of a hybrid where you've got a file which is a virtual drive hosted by one of these non-TNO services. You're providing, though, the layer of encryption, so you get the best of both. You're able to, with confidence, put files in there, knowing that they're encrypted before they leave. Yet at the same time, it's SkyDrive or Google Drive, and wow, 25GB is pretty nice for free. And then you have all of the features with none of the downside, except of course you don't get the features for those files that you have locked up in your own TNO sort of sub-vault on the service. So I'll bet that's where we end up, with a solution that makes that sort of thing work, and easy.

Terry Reck in Greenville, South Carolina said - oh, and in his little Twitter profile he said: "Mac support specialist by day, jazz musician by night." And he said, oh, he sent me a link to one of the things we're going to start seeing. This was on MacWorld. It was "Automatically encrypt files for your Google Drive." And this, frankly, isn't - it's free, but it's kind of kludgey. It uses the Mac's scripting-based Automator scripts. And when I went to go check that out, the script itself has the description: "Creates a disk image filled with files and folders from the previous action. Options include compression and encryption."

So this looks sort of like something where you manually collect a bunch of things and then drop it on this script, which compresses them, encrypts them, and then probably copies them to your Google Drive. So it's a way of encrypting stuff before they go off to Google, but it's not a seamless client that actually allows you to mount something from Google Drive and look at it like a folder on your machine, where you could easily and freely move things in and out.

Oyvind Stokke in Bergen, Norway said: "Enjoyed SN-349. By the way, TrueCrypt plus Dropbox is a great combo. Uploads diffs only. Slow the first time with large volumes, but fast after that." Okay, now, this is a very interesting idea because I confirmed that Dropbox hashes 4MB blocks of large files. So the idea being that, if you had a large file, and only some piece of it changed, the Dropbox client running on your system is smart enough to realize that only that piece of a larger file changed, and just upload that.

Well, what's very cool about that model with TrueCrypt is, if you create a large TrueCrypt file into which you create a TrueCrypt drive, then as we know, when you TrueCrypt this large container file, it will fill it with noise. Absolute, you know, it encrypts it, pseudorandom noise. That, then, you synchronize with Dropbox. And this is the part that takes a long time, if you create a terabyte volume or who knows what you're going to do. But you get it synched.

Okay, now, remember that this file, this container, internally it's mapped just like a physical drive, meaning that in the front are directories. And then there are the equivalent of file allocation tables. And the actual physical space in the file represents the physical span of a drive. So reading and writing files in and out of it would only be changing those sectors, the sector equivalents, in the file. Which would mean that things you do would only change pieces, which would then be rounded up to 4MB. And those 4MB chunks would get synchronized through Dropbox. So this is interesting.

Now, one glitch, maybe, is that I did encounter a reference to needing to unmount the volume for Dropbox to synch it, which of course breaks the whole coolness transparency of it. So I haven't tried this. Maybe some listeners will and let me know what the limitations are. But if you are a person who likes TrueCrypt - now, the downside is, and there are other solutions that don't have this problem, obviously, the downside is the whole concept is you create a fixed-size container, which means you're creating a fixed-size virtual drive. That is, it is not inherently elastic. It isn't expanding and contracting.

Things based on the encrypted file system, which we talked about last week and we'll touch on this week, they do encrypt files and the directory system metadata. So there you get an expanding/contracting sort of environment. But TrueCrypt is TrueCrypt, and there's a lot of good things to be said about that. And so depending upon this need to unmount it or not, that may or may not be the problem. And it may very well be that other of these less-secure-than-we-wish services, but who otherwise provide good features, like many people really like Dropbox, can have a TrueCrypt container file hosted on them and then really get TNO security.

Matthew Figur said: "@SGgrc, you can use SecretSync with Google Drive as it functions just the same as Dropbox for some security." And so we covered so many of these options, these cloud providers last week, I had to go back and remind myself what the heck was SecretSync. I remember it seemed kind of good. And so it is doing pre-Internet encryption and apparently is TNO. But it's one of these that has recurring charges. No charge up to 2GB, and if your needs were modest, then it's free forever at 2GB. But if you go beyond that, it jumps to $40 per year for up to 20GB of encrypted storage, and then beyond that it's $60 a year for a terabyte. So it's like, okay, maybe that makes sense.

I liked the way it looked. We'd have to really vet it to make sure that it was TNO, and I haven't. But it is one of those where you're paying them for the encryption privilege, and you're paying somebody else for the storage privilege. So it's not clear to me that that makes as much sense as a solution which is free. And my sense is we're going to be going there. There will be alternatives that are free before long.

Toby Dawson says: "Might be worth a look regarding online storage solutions, includes drive." Oh, this was interesting. Very interesting, actually. This was - The Verge hosted a huge, wonderful grid. In fact, I tweeted this recently, so you can find the link in my recent Twitter feed. It's not very deep at this point. Just @SGgrc. But I also made it this week's bit.ly link. But it's not sn-250. Something, someone probably getting up to some mischief, or maybe not, but already used that. So this week's bit.ly link is /sn-clouds, all lowercase, sn-clouds. This is a very nice feature-oriented comparison of all these things: Google Drive, Dropbox, SkyDrive, SugarSync, clouds, I mean, all these guys. And it goes through them one by one.

But if you scroll way down, about halfway down, they pull it all together in one huge grid. Again, it doesn't talk about security because of course that's a little harder to figure out in many of these cases. But it's definitely got a nice feature comparison of all these guys. So once again, bit.ly/sn-clouds, all lowercase, and you can find it. Or I just did recently tweet it, so you can find the link there, too.

Tyler McHenry, a software engineer with Google in Sunnyvale, California, tweeted. He said: "Gdrive" - obviously his drive, Google Drive - "could be TNO if used with encfs," meaning the encrypted file system. He said, "But the web UI would be useless, of course." So Tyler is saying the same thing I've been saying, and I'll bet you we end up with a solution like that, sort of a hybrid where we come up with something that's using the encrypted file system, and then it looks like a drive or a file which is growing and shrinking and gives us all the benefit of full TNO security, yet at the same time all the features of Google Drive.

Justin Gerace said: "Take another look at Box. You sort of glazed over them in your discussion on Security Now!. They have a big presence in the mobile space." Well, I looked at them again, and they are neither pre-Internet encryption nor Trust No One. And the storage costs are extremely high, about three times what Amazon and Rackspace costs. So I just don't see that there's much, I mean, yes, they were there. They've been there for a long time. They've got a lot of inertia. They've got a big following. My feeling is the future is uncertain because they're going to be undercut by - now that the big boys are getting into the game. And, boy, we've got so many options for much better security. If you need the features they have, and you don't need pre-Internet encryption, or you can use something to layer that on top, then it might make sense. But still pretty expensive.

Grahame Todd in Dublin, Belfast said, oh, he asked: "Would you be interested in SafeBoxApp.com?" He said: "It's pre-egress encryption for Dropbox, Google Drive, Windows Live SkyDrive, SugarSync, ZumoDrive, Carbonite, F-Secure, et cetera." And I checked it out, SafeBoxApp.com. It's not clear what they're going to have. They're coming soon. They offer it for Mac and Windows. You can sign up for notification. So there's just one more opportunity. And since we are running long, I think I will take your advice, Iyaz, and skip these last guys because there's nothing really crucial in those.

I did note that SpiderOak has a promo code of "Spring." So for anyone, I don't know what that means, I don't know what it gets you yet, but I just wanted to pass it along. I don't think it's super top secret. They didn't tell it to me, someone else did. So the promo code is "Spring" for SpiderOak.

I wanted to note that TweetDeck is now available for the web, a just general browser interface. I was very impressed: web.tweetdeck.com. I tried it under Firefox and Chrome, and it's beautiful. It comes up, and it's like full running Twitter TweetDeck in real-time on those browsers. So that's just way cool.

Okay. So I did have a note that I wanted to share about SpinRite from a Todd Bertels, who is a listener. He said, "Steve, good morning. I'd like to share with you some of my successes with your SpinRite app. After listening every week to you and Leo for the past year, I've decided to share some of these with you. I've used SpinRite on several laptop, desktop, and server hard drives, and found that as long as the system recognizes the drive, and there are no scraping sounds coming from the drive, it will most likely work. Most of these are unremarkable stories, with the exception of two.

"First, my server hard drive failed approximately three weeks after I installed a network-attached security camera. The camera was configured to push a still image to the server via FTP once a second and overwrite the file existing there. This file" - I'm sorry. "These images are the same size, so it wore a pothole in the drive, and the drive failed completely. After running SpinRite, everything came back except for the one camera image file. Truly amazing.

"And secondly, my wife handed me her iPod Mini when we were at the gym. It was very sluggish and would hang for three or four minutes at a time before playing again for a few more minutes. We tried synching it with the computer, but that would fail after a few attempts. It had 'failed hard drive' written all over it. So I dissected the little gadget and removed the Hitachi microdrive from inside. I went to Fry's and got a CF/IDE adapter and ran it against SpinRite. After 45 minutes the drive was running as good as new. Steve, your product is truly amazing. I've tried many other data recovery products and services over the years, and they just don't work. Usually you get data back, just enough to make you kick yourself for not having better backups. Cheers and thanks, Todd Bertels." So Todd, thank you very much.

IYAZ: Okay. So, yeah, I tried that TweetDeck for the web, by the way, just going back real fast. I don't like it. I think Twitter screwed up TweetDeck huge. I don't know. Did you like it?

Steve: I just fired it up on both browsers, and it worked. And in fact I'm a TweetDeck user. That's where I'm watching all this stuff happening all the time. So...

IYAZ: I guess I'm an old man. I like the old Adobe Air version of it, when it had, like, 15 columns. But that's just...

Steve: Well, I've got, like, eight. Is there a column limitation?

IYAZ: Well, when Twitter bought it, they kind of changed - they basically threw out all my old columns, for some reason. It was like, start over. And so then they lost me entirely.

Steve: Ah, okay.

IYAZ: Cloud storage solution. Backblaze was a cloud storage solution that you looked into extensively because apparently it has some things that need to be explained; right, Steve?

Steve: Yeah. So what happened was, in response to our listeners - we heard from some of them who were tweeting, but there were many more actually. They were saying, hey, what about Backblaze, which I had not covered last week. So I took a look at it, and it took some wading through. They've got some nice diagrams. But what you want in a diagram that explains cloud storage is you want it to clearly show you where the Internet is in the diagram. Which is to say, what is encrypted on your side of the Internet connection; what's going on on the other side of the Internet connection.

So they had some flow diagrams that talked about 248-bit public keys, and we randomly generate a 128-bit AES symmetric encryption key. Then we use the public key to encrypt it, and then that goes off. And actually I'll explain what they're doing in a minute. But the point was that, unfortunately, it wasn't clear what was being done where. They sort of had a flowchart, but not "You are here, and the Internet pipe is here," and "Nothing that shouldn't go through the Internet pipe is doing so."

So I had to dig deeper. And I ended up finding what they call their Security Question Roundup page. And so under, for example, "How good is the encryption? What do you use?" Because it's clear that many people, I mean, if these are real questions being asked, and certainly our listeners are caring because they were tweeting like crazy, saying what's the story, a question is "How good is the encryption? What do you use?" And so their answer is, "We use 2048-bit public/private keys to secure a symmetric AES 128-bit key that changes for every backup session on your computer. Usually this is once per hour. You can read about our approach here," and then they link to a "How to make strong encryption" page.

And then they say "We copied the design of Microsoft's Encrypted File System designed by a group of much smarter people than ourselves. We didn't invent any of this. It's all off-the-shelf OpenSSL library stuff." It's like, okay. And then later on on this page it says, "My place or yours? Is a separate AES" - and again, this is like people still trying to understand what's going on because it's just not made clear. "Is a separate AES encryption done on my computer before the data is sent to your server? If not, how does the key generation work?" So the answer is, "At a higher level, Backblaze uses OpenSSL. So the answer to most of your questions is found by reading up on that technology. Luckily this is the widest known encryption library and technology on Earth. At Backblaze we aren't encryption experts. We just specialize in making what is normally only for rocket scientists usable by Mom and Pop consumer." Then we have a smiley face.

Then a little bit lower down, still answering the same question, trying to, I don't know, put someone's mind at ease, they said, "The private key stored in your account in the Backblaze datacenter is also a PEM file," which is just a standardized format for securing or for storing binary data - "stored ... in the ... datacenter is also a PEM file. And by default it is secured (encrypted) by our well-known passphrase, which means technically, if a Backblaze employee was malicious enough and knew enough and spent enough time to target one of our million-ish customers, then we could have access to your file's contents." Then it says, "(a firing offense at Backblaze, and we guard the access to a very exclusive list of Backblaze employees)." Well, that tears it. I mean, what that says is that, if an employee goes rogue and decrypts their customers' data, they'll get fired. But it also says it's possible. And we know that it should not be possible.

So here's what they do. When you install the Backblaze client on your machine, they generate a 2048-bit public key pair. And we know how that works from many times we've discussed it. In asymmetric encryption, one of them is used to encrypt, the other is used to decrypt. Then for a - oh, okay. So they generate the key pair. And as part of your account setup, the private key is sent to them. And they make a big point elsewhere of saying it is never written to your disk and never stored on your computer. It's like, okay. But so you send it to them.

Then to do a backup, a pseudorandom, 128-bit AES symmetric key is generated just for that backup. The backup blob is encrypted with that key. Then that key that you used for that backup is encrypted with your public key that is uniquely yours and for which they have the private key. Then the encrypted blob and the encrypted key for it, that was encrypted with your public key, is all sent to Backblaze. And again, on their site they make a point of, and that key is never written, that temporary symmetric key is never written to your drive. It's only in RAM, and it's immediately deleted.

Okay. So what that means is that at this point they are the only people who can decrypt it. You can't. I mean, it's your data, but you can't decrypt it. So it's on their server. And when you want it back, even if, I mean - but I have no reason to mistrust them. I mean, really none. I think they're a good company, nice people, well intentioned. But they've just got a messed-up crypto model. So, I mean, if they're trying to offer really good security. And the idea now is that, if you want it back, you can't get it back because you don't have the key. You ask them to decrypt with your private key, which they have, the key that was used to encrypt the data. They decrypt it for you and send it back. So you see my problem.

IYAZ: Is there anything, I guess, after going through the FAQ here, or the Security Question Roundup, that is, is there anything positive about Backblaze after reading all this? Because it sounds like it's kind of a mess.

Steve: Yeah, it's a mess. And I wish it weren't a mess. But, I mean, there isn't another way to explain it. And also, frankly, I mean, there is this sort of sense - well, okay. What I should do now, because I - let me share with you my communication to Gleb Budman, who is the CEO and cofounder, Gleb Budman, who reached out, contacted me, wanted to know what my concerns were. I said:

"Hi, Gleb. First off, let me explain that my tweets today were triggered because so many people were asking me why I wasn't recommending Backblaze, or what I thought of Backblaze. My Security Now! podcast of last week focused upon many cloud storage providers, but Backblaze was not among them. So I simply had not looked.

"Secondly, as my most recent tweet tried to explain, I'm talking about some strict techie details, not about how and whether Backblaze is trustworthy. I have no reason whatsoever to suspect Backblaze is anything other than totally trustworthy.

"But the problem is your security model REQUIRES that your users trust you. That's not in itself a bad thing. But during the seven and a half years of the podcast we have covered cryptography and cryptographic systems extensively, and we have developed the acronym 'TNO,' which is short for 'Trust No One.'

"It is possible for cloud storage solutions to implement a cryptographic model where at no time and in no way could the storage provider possibly decrypt their users' or clients' data. And a number of Internet remote storage providers do just that. But from my careful reading of your online documentation, it seems very clear that that is not a security architecture that your organization has adopted. Thus you are not TNO safe. You can, if you choose to or were compelled to, decrypt your users' stored data."

So I then quote some of his own pages, which I won't go over again. And I said - oh, and I should mention that one of the additional complications is that they offer an enhanced level of security, which is really where this problem comes in because they give their users the option to provide a password which, if they ever forget it, they're out of luck. Unfortunately, they do this wrong. The password that the user can optionally enable encrypts their private key, which is still stored at Backblaze. So what they've done is they've layered sort of an attempt to increase the security on top of an already broken model. But that doesn't unbreak it.

So now what happens is the user who wants access to their data has to provide the password to them, which then they use to decrypt the private key, which is then used to decrypt the key that was used to encrypt the backup. So once again, the decryption is occurring at Backblaze. Only they have the key. And so they can and have made the argument that, well, but we don't - we can't decrypt it all the time. We have to wait for the customer to ask for the data. Then we can decrypt it. It's like, okay, that's still broken. I mean, all they have to do is change their model. Don't do this "we're holding the only key in the universe that can decrypt our customers' data, but we promise not to look at it." Okay. Anyway. So continuing with my note to Gleb, I said:

"From studying those pages, I understand that, one, when the Backblaze client is installed, a 2048-bit asymmetric key-pair are created. The user's client computer retains the public key, and the private key is sent to Backblaze. For every backup session, a one-time 128-bit pseudorandom symmetric AES key is generated. That symmetric key is used to encrypt the user's data, and that key is encrypted with the user's public key. The symmetrically encrypted backup data and the asymmetrically encrypted encryption key are then all sent to Backblaze."

So just to stop for a second, I mean, yes, this is confusing. And you could understand why people who just want to know is this TNO or not can't figure it out. I mean, and the diagrams aren't clear. And it's sort of, unfortunately, this is all kind of a little glib on their site. It's just not science. It's, oh, well, we'll fire them if they do that. I said:

"So this is unbelievably broken. Even the user cannot decrypt their own data. Only Backblaze, who maintains the user's matching private key, has the ability now, at will, to decrypt the encryption key used to encrypt the data, then decrypt the data. And presumably that's precisely what you do whenever the user wants to browse or retrieve their files. However, you could also do it anytime you desired.

"Now, I know that you provide a mechanism for allowing the user to encrypt their private key on your servers so that even you can't decrypt their backup encryption keys. But it's never possible to layer more bad security onto a fundamentally broken security model to fix it. In this case, if you were under court order to provide a user's data, you would merely need to wait until that user did wish to access their storage and thus provide you with their secret key to decrypt their private key. And you would then have access to all of that user's stored backup data."

Anyway, that's their situation. So Gleb slept on it overnight and sent me a note in the wee hours of the morning saying that he would respond when he'd had some sleep. And he said:

"Steve, thanks for the detailed email. I believe you are largely accurate" - oh, and I have his permission to share his response to me with our audience on the podcast. "I believe you are largely accurate in your understanding of how the system works, but are largely wrong in your resulting conclusions, with one exception which is a fair concern, and I'll address. Let me explain.

"First, Backblaze has two models for security: Trust Backblaze. In this default option we have the ability to decrypt your data. This model exists for one reason, so that users can recover their password. The point of backup is to protect people from data loss. For the majority of users, the biggest risk is that they will forget their password and never be able to recover their data. To be clear, even in this model all data is stored encrypted, and only a very few people in our entire company are trusted with the keys. So while it is 'Trust Backblaze,' the encryption still serves a valid purpose."

Now, let me pause for a minute. It's like, I completely agree with that. We all get it that Trust No One means no one but yourself. That is, if you are the sole key holder to data stored in the cloud, you can't lose those keys. I guess my argument is how hard is that to prevent? How many things do we have, tools do we have that help us remember our passwords? And there's also good old paper and pencil, if something is really crucial. Anyway, going on, he continues:

"Trust No One. In this option a user can choose to encrypt their private key with a private passphrase. In this model, neither Backblaze nor anyone else except the user can ever decrypt users' data unless the user gives them the passphrase." Well, as I said, unfortunately, the user does give them the passphrase any time they need visibility into their data. And at that point Backblaze decrypts the private key and has access. So, broken. Anyway, I don't want to criticize his response too much because he may - I want to allow him to have his say.

"While 90 percent of our customers choose the first option, and I firmly believe that for the majority of users this is the right decision because their biggest risk is not a government subpoena but losing their password. However, for the purpose of our discussion I will focus on the second option since it would be the one you and some of your listeners are probably most interested in.

"You say that our security model for storing the private key encrypted by the user's private passphrase is inherently flawed. However, this is the model that Microsoft Encrypted File System is based on." Okay. "I believe this is the same way your sponsor Carbonite does their encryption." Okay.

"If you believe us storing a private key encrypted with a passphrase is not secure, this would imply encryption inherently is unsafe, in which case it does not matter who has the private key, since all of these systems inherently rely on encryption." Okay, well, that's not correct. I mean, the problem is, as we understand now, the user gives them the private key to decrypt their - the user gives the private passphrase to Backblaze, which then allows Backblaze to decrypt the asymmetric private key and then decrypt the backup. So there's nothing wrong with encryption. It's that their model is broken. Where things are and where they're done is not Trust No One because - okay, well. He says:

"When the user picks a private passphrase, Backblaze can never decrypt the data, regardless of any government subpoena." Well, that's not true either. The model requires that they decrypt the data. Okay. So he says:

"The one exception, which I want to be fair to you about, is that when the user does a restore" - oh, here it comes - "Backblaze needs to provide the user with their decrypted data." Mm-hmm. In this model, unfortunately that's true. "Thus we ask the user for their private passphrase at that moment. There is a moment of risk here, but very short." Yeah. On the other hand, computers are very fast. "The system is automated. No employee ever sees the key. The key is only stored in RAM and is never written to disk."

Okay. So to be fair, I think these guys have done, like, the best job they can with their broken model. It's still broken. It's still wrong. And there's no reason not to trust them. But it's just, you know, it's not done right. He says:

"Going further, I claim that, if you believe a real Trust No One policy, you cannot use Jungle Disk, Carbonite, or any other provider because to use them you have to trust them." Then he has a number of points: "They have code that runs on your system connected to the Internet. They do not open source their code; and, even if they did, would you review every single line of their code every time they did a release? They can say that they do all sorts of things, but they could simply put a keylogger on your computer and send all your data directly to the government. In fact, if they're required by a subpoena to get your data, and they have an application running on your system, they could change their app just for you. Thus, even if some expert reviewed the code, that will not guarantee that the code on your computer is doing what they publicly say. Thus you are inherently trusting them to do what they say."

Okay. Well, right. All of those things are true for everybody, for all users of all of these things. Yet there are solutions whose architecture, by design, keeps the encrypted data on the local machine and only ever ships out decrypted data and never discloses its keys. And that's not what this does. And he says:

"What we say: We encrypt all of your data by default on your computer. We send all that data over an encrypted connection and store it encrypted. You can choose to have your data encrypted with a private passphrase of your choosing. If you do, NO ONE" - he has in all caps - "can ever decrypt your data unless you provide your private passphrase." That's true.

"If you do a restore and enter your private passphrase, the key is stored in RAM, not written to disk, not seen by anyone." So he's saying, given their model, they're trying to be as responsible as they can. They're doing the best job they can, which I completely accept. He says:

"Furthermore, the moment after restoring your files, you can change your private passphrase. Thus, again, it would be impossible for anyone to decrypt the data, even if we wrote new code to capture your key, which we don't have." And of course understand that to decrypt your files, they are the ones who decrypt them, which is fundamentally the problem here. Nothing can get around that. Then:

"We never decrypt data without the user explicitly requesting it, or if a government subpoena required it." Okay, well, that's comforting. "And with your private passphrase, that is impossible. In the five years of Backblaze, we've never been subpoenaed or decrypted any user's data without their permission." And of course they have to say that because they're legally obligated not to disclose, under Patriot Act, if they've done so. So he volunteered that, and I believe it, but that doesn't mean it won't happen tomorrow. So he's finishing:

"Should you trust us as a company to do what we say? We're very public about who we are and are accessible on email, Twitter, Facebook, Google+, both as a company and individually." And they are. "We've been in Silicon Valley building products and companies for over a decade." And they have. "Our last company was an email security company, where we protected some of the largest companies on the planet from spam, viruses, phishing, etc.

"Does that make sense? I'm happy to chat, connect you with our CTO or VP of Engineering, share more details about our approach and systems, join you on your program, or anything else you would like." So I thank him. And I think everyone understands now. This is a great example of an attempt at security. I mean, there is lots of security here. Data is encrypted. It is always encrypted as it goes to them. Unfortunately, their model, and I don't understand who designed this, I mean, they earlier, and they've said on their public pages, they're not cryptography experts. And I'm fully going to agree with them on that, too. This is just not correct.

And my argument, and I responded to his note, and I said, look, I'm not telling you what to do. You can obviously do anything you want. And I accept that you're good guys. But there are people who care, and they believe something that is not true from reading your pages. It takes really carefully understanding what you have said in your technical documentation to get it. And the reason there's been some concern generated when I posted on Twitter that this is not a Trust No One architecture, is that people thought it was. The pages are so confusing, I mean, I don't want to say deliberately so because they're sort of like, oh, well, you know, we're not rocket scientists, and neither are you, so - and besides, we used OpenSSL and we copied Microsoft, as if any of that means anything.

So I think these guys are fine. I think they're trustworthy. And it is true that, as long as things are static, if you employ their optional passphrase, then your copy of your private key is encrypted on their system, and they can't get your data. It's not until you want to look at it or ask for it that you need to provide them the ability to decrypt your data and send it back to you. But that's just not the way it should be, not if they really want to provide security. And I did write this in my response to him. I said, all you have to do is not send the private key over, just send back the encrypted data and let the user decrypt it himself. So easy to fix this, but it's not the way the system works now. So that's the story. That's the story.

IYAZ: Do you think they're going to implement your changes? Since you just gave them free advice.

Steve: There was, well, and - who knows. I saw someone tweeted something about a 2.0, but this wasn't coming from Backblaze. And a response did indicate - I think he responded to my response, which thanked him for his clarifications on things. But there was something about moving forward in the future we're going to make things better. So, okay, that would be good.

IYAZ: Well, hopefully they'll at least clarify their security questions, and at least there's documentation about it. Because it does seem like they're just trying to give you some reasons. You're hoping they can figure it out. But thanks to you, Steve, everyone knows now. So it's a lot easier. Wow, that's - because I read that. I read that documentation, too. And I was just like, yeah, it seems a little bit like they're being honest, but seems like they don't exactly know.

Steve: Well, yeah. They seem like, I mean, I really get the sense these are good people. But they're not crypto people. Maybe they're email people. I don't know. But, I mean, it's just a dumb architecture. It's like, I mean, it's really - you generate a random key to encrypt your data, and then you encrypt that with your public key, and the private key that matches was sent to them, and they make a point of telling you it's not on your computer, it's never been stored. Which means you cannot decrypt it. Your own data. Only they can. It's like, okay, who came up with this? Just nutty.

So again, but they're offering way more security, for example, than SkyDrive and Google Drive because those guys aren't doing any client-side encryption. And there are all kinds of, you know, their privacy statements say that the data is safe in the cloud, and it's encrypted, and blah blah blah. But there is only - there's absolutely a simple test, and that is - for TNO. And that is decrypted data never leaves the machine. Only encrypted data ever exits the machine. And keys for that never leave your control. They also never leave. And that's all there is to it.

If you have that - and, see, the beauty is, here they're, like, worrying about having to fire people and needing to lock down access to the key that could decrypt. Remember that 90 percent of their users think they're being encrypted on their machine, and that that means it's, like, safe at the other end. Except that they just said some employees have access to the key that can decrypt 90 percent of, right now, of their customers' data, all those people who didn't use the extra passphrase. Which means that they've got all this responsibility. Why have that responsibility? If you architected with TNO, you don't have to worry about your own employees or government subpoenas. You can make a much more simple declaration of your security, rather than these pages of gobbledy-gook that nobody really understands. Just say no, no one can get it, period. And it's true. Unfortunately, it's not here.

IYAZ: Well, thanks, Steve. It's been an education. And I hope I didn't get the show canceled. But it's been just a lot of information in my head. You can find all Steve's stuff at GRC.com. ShieldsUP! is a fantastic program. And if you want to know about the steps we talked about in this show, you could check out the show notes at TWiT.tv/sn. And thank you so much, Steve.

Steve: Iyaz, it's been an absolute pleasure. And now I've got two backups for Leo who are qualified to do the podcast with me. So I'm glad to have you, thanks.

IYAZ: I'm glad I survived the baptism there. So thank you, Steve.

Steve: Bye-bye.