SECURITY NOW!

Transcript of Episode #376

Fully Homomorphic Encryption (FHE)

Description: This week, after failing to find much in the way of interesting security news, Steve and Leo make up for that by introducing the concept of "Fully Homomorphic Encryption," which allows encrypted data to be operated upon WITHOUT it first being decrypted, and results remain encrypted.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-376.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-376-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson, the Explainer in Chief, is here to talk about - get ready. Get your propeller beanies on. This is one of those special Security Now! episodes. He's going to explain a new kind of crypto, invented in the late '90s, so it's practically brand new. Fully Homomorphic Encryption, next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 376, recorded October 31st, 2012: Fully Homomorphic Encryption.

It's time for Security Now!. Get ready, ladies and gentlemen. Propeller hats at the ready. Here he is, our Explainer in Chief, Steve Gibson. We talk, as always every week, about technology, about protecting yourself, about security, about privacy. But sometimes we get into the weeds. And I don't know, but I'm just looking at the title of today's show, and I think this may be a weedy episode. Hi, Steve Gibson.

Steve Gibson: Well, you know, Leo, we're not going to have you for three weeks. Tom Merritt and I are going to be doing the next three Security Now! episodes. So I thought we would leave on a high note, or a wound-up propeller note or something. We tackled ECC, Elliptic Curve Crypto, two weeks ago, of course. And I thought, while we're on the topic of esoteric encryption, we ought to take a look at where the crypto world is today. Everything we've always been talking about is kind of - it's new for us, and it's in use now, but it's kind of old and dusty. I mean, all this stuff is like, you know, the RSA patents expired a few years ago, so they're 20 years ago. And even the just-chosen SHA-3 secure hash algorithm, that competition began eight years ago, in 2004. So what's going on right now? And one of the things going on right now, one of the things that cryptographers are actively working on is something called "homomorphic encryption."

Leo: Okay.

Steve: Or "fully homomorphic encryption." And I'll just, before we get into our top-of-the-show stuff, I'll just - I can explain what it is, that is, and what it accomplishes. And it'll just, like, bend our brains.

Leo: Yeah.

Steve: It is possible; it has now been shown. It was hypothesized by the RSA guys back in '78, shortly after they came up with the whole "factoring is hard" public key encryption technology. They posited the possibility of this, the possibility of homomorphic encryption. But it was an open question. And just three years ago a grad student stunned the world by doing it. And what he did was mathematically demonstrated that it is possible to perform operations like standard computing operations - addition, multiplication and so forth - on encrypted data without first decrypting it. So you encrypt data...

Leo: Wait, wait, wait, say that again.

Steve: I know. It's amazing.

Leo: But what?

Steve: So, for example, you could encrypt data, send it to the cloud, have work done on it, never decrypting it. The cloud has no idea what it's doing. I mean, it knows what the processing is it's doing, but it never sees any result. No intermediate results. The all of everything stays encrypted the whole time. You get your result back encrypted, and only you are able to decrypt the result. And this works. So that's the topic of today's podcast.

Leo: That's exciting. I don't understand how it could possibly work. But we'll have to figure it out.

Steve: It's extremely cool.

Leo: And I know we'll find out.

Steve: Yes.

Leo: I was right. Propeller hats at the ready.

Steve: Oh, boy.

Leo: Oh, boy.

Steve: And that may be your costume for today's Halloween podcast.

Leo: I should point out neither Steve nor I are wearing Halloween costumes. I wore a trench coat in. But it's because it's going to rain. It's a raincoat. It's not - I'm not - I should have just said, oh, yeah, I'm Inspector Gadget. There, Steve's got his TWiT fez on. That's a good costume. No propeller on it, however. Nice tassel. Our show - no, you follow the - you're like a kitty cat. You can't - a dog chasing its own tail.

Steve: It's a homomorphic tassel.

Leo: All right, Steve Gibson. We have a little bit, not a lot, but a little bit of technology/security news before we get to homomorphic crypto.

Steve: I scrounged around looking for interesting things that happened.

Leo: Quiet week.

Steve: And, yeah, the only thing I could find that I thought would be interesting is in our ongoing following of the infamous Do Not Track header that we've talked about for a couple years now.

Leo: Yeah. And the last I heard, Microsoft is still saying IE10 is going to turn it on by default, which means Apache will ignore it.

Steve: Yeah, in fact, IE10, I guess, already happened.

Leo: Yeah. We have it.

Steve: And it did not get pulled at the last minute. So not only Apache, but now Yahoo! has formally stated that they intend to ignore it. Their term for it is "signal abuse."

Leo: Oh, wow.

Steve: Since the W3C Consortium said that it is to be off unless users turn it on, Microsoft, feeling that users would turn it on if they understood what was going on, have it on unless users turn it off. And that's enough for Apache, as you said, and now we know Yahoo!, to say, okay, we're just going to ignore it.

Leo: Apache maybe is a little controversial. But Yahoo! is an advertising company, basically; right?

Steve: Yes, yes.

Leo: So that's not too...

Steve: And what's interesting is that Microsoft is planning to make IE10 available for Windows 7, and they intend to still leave it set on unless it's turned off by the user at installation time.

Leo: Wow.

Steve: So this will be an evolving story. I'm proud of Microsoft. I don't say that. Have I ever said that? I don't think so.

Leo: Wow. What?

Steve: They never do anything that I think they should, when they should. And they've, like, I'm really impressed. This is great.

Leo: You're impressed that they are turning it on.

Steve: Yes.

Leo: But you're not impressed with Yahoo! or Apache for ignoring it.

Steve: Not at all. I think they're on the losing side. I think this will be a skirmish for a while. It will end up being decided in favor of the user, which is obey DNT, and IE may get a reputation for being more private because it's got this turned on by default. And maybe other browsers will follow suit. I mean, we'll have to see how this happens. It is, for me, sort of an interesting little political sidebar on the whole "how our security industry is evolving" story.

I'm under embargo for some news that I cannot talk about until next week's podcast. I will say only, I can say only that there will be a significant update to one of our favorite TNO, Trust No One, cloud storage offerings that I will be able to share with our listeners, and I'm excited about it, a week from now. And that's all I could find that was...

Leo: And there's the news.

Steve: And there you have it. Sorry about your commute, listeners.

Leo: Well, we're going to get to homomorphic crypto. And if you are driving the car, some of you, you will find this stimulating, and we'll wake you up. Others you might want to hold off until you get home. That's all I'm going to say.

Steve: I actually think this is going to be good. I did want to thank a number of people who said, hey, Steve - this is over the last couple weeks - what happened to the blinking lights behind you? Because...

Leo: Wait a minute. They're gone?

Steve: They are, again. But they haven't been for a couple weeks.

Leo: Oh, I didn't notice.

Steve: And I realized, well, we'd had a brief power outage a few weeks ago, and all three of those little PDP-8s just came to a standstill. And I'm so used to them blinking that, when they weren't, I didn't notice it. But people were looking at the video going, hey, the lights have stopped. And so it was when they said, "What's going on?" I said, oh, that's true, and I started them up again.

And I just didn't want to - I wanted to do a shout-out about the TV series on Showtime, "Homeland."

Leo: Oh. I haven't watched it yet. Is it good?

Steve: Oh, my god.

Leo: Now, the premise of this is what?

Steve: Okay. We're in the second season.

Leo: This is like Manchurian candidate, kind of; right?

Steve: We're about four - yes. We're about four weeks in. But last Sunday night's acting - Claire Danes is one of the leads. And some other random guy who we've never seen before, but he's good, it was an interrogation, pretty much, for most of the hour. But knowing all the back story and all of what was going on and very complex characters that have been painstakingly created, I mean, and she deserves awards, I think, for the job she's doing. Anyway, you can get the disks or find it or borrow it or something for the first season. You really do need to catch up. But it is just - I don't know where it's going to go. But it has been a season and a third of real fun. I mean, just...

Leo: No spoilers now. No spoilers.

Steve: No. I'm not. I don't do that.

Leo: No, no.

Steve: It is really, I think...

Leo: Because I haven't watched any of it. But so I should start at the beginning with season one.

Steve: Oh, you have to. Have to start at the beginning, get the first season. You'll absolutely be hooked. And this second season is every bit as interesting. And I have to say, after the first season ended, it's like, okay, now what are they going to do? And it is certainly the case that a concept can outlive its production. We saw, for example, "Galactica" went off the rails, like in about season 4 or something, or 3.

Leo: It's hard. Four seasons is hard for any show. Even "West Wing" kind of went downhill after four seasons. It's hard.

Steve: Yeah. Then you have the flipside, where like "Firefly" gets canceled before even the ones they've produced have been aired.

Leo: [Growling]

Steve: Okay, yeah.

Leo: So I'm looking, and "Homeland" is not on Netflix. Let me see if it is - Instant Video, Amazon Instant Video has it.

Steve: Oh, good.

Leo: Or you can buy a DVD. But I'll just screen it on Amazon.

Steve: It's really, really, really worthwhile.

Leo: All right. I'm going to watch it. I'm going to start tonight.

Steve: You'll get sucked in. It is, I'll just say, and this is not any spoiler, she works for the CIA. She's a field agent with expertise in the Middle East. And she is also, and this is not a spoiler, she is bipolar, medicated, but secretly because she could never work for the CIA if they knew.

Leo: Oh, that's interesting. How interesting.

Steve: Oh, and she's playing her character so well. I mean, it's just - it's really good. So for what it's worth, if anybody has some time to kill, or likes...

Leo: Damian Lewis always reminds me of Steve McQueen. He has that pursed mouth, that Steve McQueen mouth.

Steve: Yes, yes.

Leo: All right, Steve. I'm ready for homomorphic...

Steve: We'll do that in a second because I wanted to share a fun story...

Leo: Oh, yeah.

Steve: ...that I was sent by a listener, James V. in Northamptonshire, England, who caught my attention with a subject like "SpinRite produces the evidence." Now, you can probably already guess what that's going to be about. But the details are interesting. He says, "Hi, Steve. SpinRite saves the evidence. Or, if you like, SpinRite catches the criminals. We use a lot of standalone video recorders on our customers' sites." He's with a security company in the U.K. He said, "As you well know, this application treats hard drives very hard, as the 400 H264 images every second are constantly written to the drives."

Leo: Wow.

Steve: 400.

Leo: 400 a second.

Steve: Maybe he's got multiple cameras. So anyway, he says, "So we get a call from a customer of ours saying that there's been an incident, and the police are wanting a copy of the images from our security camera system to permit them to investigate. So an engineer is duly dispatched to the site. No broadband access for security reasons." Of course he's a Security Now! listener, so he understands, you just really can't connect, period. And he says, "On arrival, I receive a worrying phone call from the engineer saying that the DVR is, quote, 'Hot enough to fry an egg on and doesn't work properly.'"

Leo: That's not good.

Steve: "Oh, dear. The machine is swapped out and brought back to the workshop where we confirm that the machine goes no further than the boot loader. We poke, prod, and mess with the drive connections until we get the unit to boot. But guess what. The evidential images we need are on the drive we had to disconnect in order to get it to boot at all. One conversation with our IT guy, and he suggests SpinRite. Now, having listened to the Security Now! podcast, I know how SpinRite has saved the day and saved the pizza. But can it save the CCTV? The HDDs are formatted in some form of proprietary Linux configuration by the machine, which locks down the part of the disk used to make it unreadable out of the DVR, both under Linux and Windows. So we were a little unsure, to say the least.

"The IT guy arrives, plugs the HDD [the hard drive] into a spare desktop and cranks up his copy of SpinRite. The drive makes some strange sounds." He says in parens, "(Bearings?)" Then he says, "Next morning, the drive was a bit warm, and DynaStat had done some business. I think it said it had recovered some sectors. So we quit out of SpinRite before it had finished and remounted the drive. Lo and behold, we could access the images, which were quickly exported to DVD.

"I understand that, thanks to the recovered imagery, the criminals were apprehended and had no choice but to plead guilty in court, both receiving custodial sentences. The world is truly a safer place with SpinRite. Without it, there would have been no evidence or conviction. We also have upgraded maintenance contract with the customer which includes twice yearly HDD checks with, you guessed it, SpinRite. I'll be purchasing a copy of SpinRite on behalf of our customer today, a bargain at around 60 of our GBP.

"Thank you for SpinRite, and also thank you and Leo and Tom for the great podcasts. Proof now exists that you make the world a lot safer place. James, Northamptonshire, England."

Leo: Northamptonshire.

Steve: Shire.

Leo: Shire.

Steve: Are you shire?

Leo: I'm shire.

Steve: I'll bet you are.

Leo: [Laughing] But you don't say "Worcestershire" sauce, do you? You say "Worcestershire."

Steve: You're not going to be shire of anything after we get through with homomorphic...

Leo: No, I'm ready to have my brain scrambled.

Steve: Okay. When you're talking about homomorphic encryption, you run across sentences like "The decisional composite residuosity assumption is the intractability hypothesis upon which this cryptosystem is based."

Leo: Absolutely.

Steve: I know.

Leo: I couldn't agree more.

Steve: Put that on your T-shirt.

Leo: You know, they could write that more clearly. I think that's intentionally turgid.

Steve: [Laughing] Well, okay. So RSA Lab's glossary definition says, of something called "probabilistic encryption" - this is something we've never talked about before. Everything we've ever discussed has been deterministic encryption, meaning that, whether it's a symmetric cipher where, under the influence of a key, you put something in, and you get something different out, and there's a way to reverse that. The idea is, every time you put the same thing in, you get the same thing out.

But early researchers in encryption a few decades ago were concerned that that weakened the system. That is, if you always got out the same thing for what you put in, didn't that coupling create some weakness in the system. So what was - and this is a couple decades' worth of work. There was this notion of probabilistic encryption where your actual encryption algorithm would produce a different result every time you used it with the same data, so that it broke that deterministic aspect deliberately. And so even if the same data was encrypted, you would get different results.

Now, we know, those of us who have been really paying attention will know that we've solved that problem in other means. We have the notion of a cipher, like Rijndael, the AES cipher, or any other cipher. But then what we've done is we've created these block chaining modes where we take a so-called initialization vector, an IV, and that is given a pseudorandom value which can be known. It's okay if we know it. In fact, often it's like the first, it's the start of the message, is here's the initialization vector under which we're going to encrypt this block. And then you take that and mix that with your input, then encrypt it. Then you take that encrypted output and mix that with the input of the next block and encrypt it. In other words, you link these together sequentially to create a chain.

So that's the way we solve the problem, which truly was a problem in terms of information leakage because, if you encrypted the same thing and got the same result, then even if someone didn't know what the input was, if they inspected lots of outputs, they could see when things were the same and begin to draw conclusions.

Leo: So this limits that problem. But how is it reversible? That's what I don't get. Because you don't know - all right. I'll let you explain.

Steve: You're right. It is tricky. Okay. So a whole different approach is probabilistic encryption. So the RSA Lab's glossary says, "Probabilistic encryption is a design approach for encryption where a message is encrypted into one of many possible ciphertexts, not just a single ciphertext as in deterministic encryption. This is done in such a way that it is provably as hard to obtain partial information about the message from the ciphertext as it is to solve some hard problem. In previous approaches to encryption, even though it was not always known whether one could obtain such partial information, it was not proved that one could not do so."

Now, okay. What that meant is that, notice that in all of the crypto we've ever talked about, we've relied on an assumption. That is, for example, it has never been proved that standard RSA public key crypto is safe. It relies on the presumed difficulty of factoring large primes. But no one's ever been able to prove that it's hard to factor large primes. We just know it is. But knowing something and proving it are worlds apart in terms of academic crypto technology. So what we've done by stepping from the world of deterministic encryption with presumed security is we actually have provable security for the first time. I mean, for the first time. That's one of the things this gives us.

So what happened was, shortly after the original RSA guys invented this factoring-based asymmetric encryption, they noticed that there was a property that their approach had, this so-called "homomorphism." And they wrote, a few months after developing RSA, about sort of this - they posed this question. Now, homomorphism, look, if we just break the word down, "homo" and "morphic," that means same shape, essentially. And the concept is that you can apply different processes to the same data and get the same result. A simple example would be, for example, the way we know that we can multiply A and B to get a result we'll call C. But we also know we can add the logarithms of A and B in order to get the logarithm of C. And so those are homomorphic operations, that is, A times B equals C, and log A plus log B equals log C.

So what happened was this idea just sort of sat doing nothing until just three years ago. And one of the cool things about this is now we're talking about state-of-the-art, leading-edge, bleeding-edge crypto, which has got everybody excited. Just three years ago a doctoral candidate, a grad student at Stanford named Craig Gentry, wrote a PhD thesis. And, boy, did he get his doctorate. IBM Research snapped him up immediately. So this was in 2009. He stunned the crypto world by laying out a fully working, fully homomorphic crypto system.

So what that means is, as I said at the top of the show, is he demonstrated - and I'm going to explain enough of this, very much in the same way as I did two weeks ago with elliptic curve crypto, that we'll sort of have a conversational knowledge, sort of a conversational understanding of how this works. I mean, we're not going to go write the code. And, well, for lots of reasons. There are many complexities to doing this.

But what Craig demonstrated in his doctoral thesis was that it was absolutely possible to perform addition and multiplication operations on encrypted data where, at every stage of the way, the data remained encrypted; the result was encrypted; and no one doing that work ever gained any information at all about the nature of the data they were doing the work on. And when the answer then was returned to the person with the key, they were able to decrypt it. And the result after decryption was exactly the same as if the same operations had been performed on the unencrypted data.

So this is huge for the future. This allows - as I'm reading, like, people talking of dreaming about applications. I mean, for example, corporations could sub out work on their data in the cloud, leaving it fully secured at all times, and have data processing done on the data with never having to trust, I mean, this takes TNO and squares it because they never have to trust anyone they hand their data to. They can have it processed and returned. There's actually - the cryptologists have designed search engines where your query is never known to the search engine. The search engine doesn't know what results it finds for you, yet it sends them back. And you then are able to decrypt them and get the results of your query with total privacy. And the really interesting applications are in the utterly tamper-proof electronic voting sphere, where it is possible to get anonymity and absolute tamper-proofing in a homomorphic encryption setting.

So as a consequence of the fact that this is sort of on the leading edge of what's going on, and that I wouldn't be at all surprised if we start hearing more about this in the future, I thought that now would be a good time to bring everyone, sort of give everyone a sense for how this works.

So we need to step back and create some analogies that I will then use to move us forward. Back in the early days of computing, before digital computers, we had analog computers. And they were patch boards of analog functions. Like you could have an adder, where you'd have, like, two voltages or two currents, depending upon whether this thing worked on voltage, whether it considered voltage to be the thing that carried the value, or the current carried the value, but whichever. It would take two of those and sum them and then produce an output that was equal to the sum of the two input values. And you could have a divider where you would put in the two values to be divided, and you would get the result from that. You could integrate. You could basically perform all the standard operations we're familiar with in an analog environment.

Now, one of the problems, if you patch together a really sophisticated equation, which is what early analog computer pioneers did, one of the problems is that you would accumulate errors because, as your inputs moved through more of these stages, each stage introduced some error. Could be really, really small, but still it was non-zero. Just due to, for example, temperature in the room could cause some drift in the amplification of the adder, or even though they'd worked to trim the component tolerances to be exact, the two inputs to the adder might not have exactly the same signal strength in effecting the output.

So, well, and even simple components like resistors, capacitors, transistors and, back in the day, tubes, there was something in tubes called "thermionic noise." But just thermal noise, just actual noise from the physical electron movement in the devices introduced some errors and noise. So the problem was this was additive as the signal moved through a patch board of these modules, additions and multiplications and integrations and divisions and so forth. And so there was a limit to how much computing you could do before you had a problem with the famous signal-to-noise ratio, that is, the signal being the actual result and the noise being the uncertainty created by the fact that this was all sort of the best we could do, but not certain.

And in fact it is arguably the fact that digital systems don't have any of this problem which has allowed digital technology to take off the way it has. As we know, in a digital system, as compared to an analog system, in an analog system we've got a continuously varying voltage. In a digital system we decide instead we're going to tolerate some imprecision in the exact specification of a value. That's known as quantization. In return, though, we're going to get absolutely noise-free processing. Noise cannot creep in because at every stage we're dealing with either, famously, a one or a zero. And each stage of our system sort of reasserts the one-ness or the zero-ness so that even values that are not quite one or zero, when they're being put in, they come out strongly one or zero on the output.

So, okay. So I wanted to kind of create that picture in everyone's mind of a network of processes that are a little noisy, and the depth of the network that you can create is limited because at some point too much noise accumulates. Because, bizarre as it sounds, this bears directly on probabilistic encryption.

Imagine a simple cipher, a simple encryption where you have a single dimension. Think of it maybe like a rope with knots along it. And these nodes, or knots, represent values that are well understood because of their position. And that the act of encrypting is choosing a node along this line and then deliberately adding some noise. That is, shifting the location a pseudorandom amount away from the proper location. So we deliberately add some noise to the location that we're choosing for a value.

Now, the key in this system, the cryptographic key, determines where these nodes are located. They're not all uniformly positioned. There is a complex calculation for where these nodes are located so that only somebody with the key knows where the nodes are. Now, we have this notion already of addition from, like, you take two values, and you sum them. So imagine that we take, we create two of these quantities with some noise, and we add them together to get, like, on this timeline, on this linear scale, to get their sum. Well, that's going to fall somewhere. And notice that the noise that we added to each of the terms being summed, the noise sums, as well. So that our final position will be a function of both of the input terms and both of the noise terms, the pseudorandom sort of fudge factors, the noise that was deliberately put in.

Well, as long as that's not too much, as long as there's not too much noise, our sum lands on this scale. And if this is sufficiently large, and we have a sufficiently large resolution, then that result doesn't mean anything to the person carrying out that operation. That is, all they know is they received a couple values, and they added them, but there's no meaning to it because the meaning is a function of where it falls relative to the nodes, sort of the location markers on that scale. And that is only known to the person holding the key.

So you sort of see how it's possible to, by deliberately adding noise and having a scale which is not known to the person doing the work, but which is known to the person receiving the answer, that it's possible to sort of subcontract the work of doing the addition. And the person doing it knows they've added a couple numbers, but they don't know anything about the actual underlying data. Well, that's a 10,000-foot sort of sense for how this probabilistic encryption operates.

The actual work that is done is not done on a one-dimensional line, or even a two-dimensional plane, or even a three-dimensional space. It's actually done in abstract algebra called a "lattice." And these are N-dimensional interconnected spaces where the dimensions are, like, 512 dimensions, or 2048 dimensions, or larger. So they're something you can - we can visualize a cube in three dimensions, and you can envision, okay, like in four-space you'd have families or sets of cubes at the different - in a series of node locations and so forth. So mathematically you can represent this, even though it's arguably rather difficult to visualize it.

And so the way these systems actually work in these lattices is that the work being done is moving a point through this hyper-dimensional lattice with noise so that the processes that are available are addition and multiplication, although it's been shown that we can do anything that we want to with just those two operations. So that's, from a theoretical math standpoint, that's sufficient. And the problem is that this system that I've described so far is homomorphic in that it satisfies that criteria. But the problem is this noise accumulation because, as you do operations on data that is deliberately noisy, as I said in the case of addition, you are doubling the noise when you add two factors together. In multiplication, you are squaring the noise when you multiply two factors together. So noise gets out of hand very quickly, and very much in the same way as with an analog computer, where you can only go so many stages, and the noise begins to overwhelm the signal.

The way this system works is, once we're done processing, and we're at some location in this hyper-dimensional lattice, the answer, that is, the actual decrypted result, is the node we are closest to. And so you can see that - and that's going to - and so our movement through this N-dimensional space has been deliberately noisy to obscure any actual values. And we need to then determine which final node in this 8,000-dimensional lattice we're physically closest to. So the point is that, as we do these processes, the noise accumulates, and that limits how much work we can do.

Well, a fully homomorphic system has no limits. That is, by definition, the definition of a fully homomorphic encryption system is one where you can perform any operation, that is, of arbitrary complexity. And this is Craig's invention. I mean, this amazing insight he had was he said, wait a minute, is it possible to perform a reencryption of the data in this process? That is, essentially, can this homomorphic encryption perform its own encryption? Because, if it can, that is, if there's time to perform this encryption without its own noise overwhelming it, then it's essentially able to reencrypt the data and zero out the noise. And initially he was unable to do it.

But by using much larger word lengths and a more complex topology, he was able to trade that off for the number of operations needed such that he was able to perform a reencryption of the data itself without noise getting in the way. And as soon as he was able to do that, he had a fully homomorphic encryption system that can perform any operation that we know of in computation while keeping the data encrypted, and never allowing this noise to get out of control because, after some number of steps, then essentially the data is reencrypted, never returning it to plaintext, but sort of re-zeroing out the noise so that it never overwhelms the system.

And, now, to give some sense for why we're a ways away, his fully homomorphic encryption - he has four classes of system sizes. First of all, I should say that IBM Research snatched him up, and he's continuing to work on this. It's been three years since his paper was published. After it published, it shocked the crypto world and got everybody excited because they didn't know what they were going to do for their summer. And they began playing with this. So there have been lots of variations, lots of ideas. I mean, basically this is an active area of current crypto research.

So at IBM he has built one. He has implemented this. And he was originally thinking that he would use one of the IBM crazy Watson hyper-computer deals. Turns out they were able to implement this on a much more modest system. He has four sort of scales of the system. He has a toy system which is - he calls it 2^9, which is to say, it is a 512-dimension lattice. A small one is 2^11. A medium one is 2^13, and a large one is 2^15. And that being, you know, beginning to be practical security.

But to give you a sense for why we're a ways away is implementing this system on standard technology computers, and this is why I'm not worried about it in the long term, we're going to fix our computers, if we're interested in this, to do it. But, for example, the somewhat homomorphic encryption system, not the fully, but the somewhat, just in the toy implementation, using a 512-dimension lattice requires a bit length for its processing of 200,000 bits. So incredibly long word lengths. The public key used in the fully homomorphic encryption system is 17MB in size, and it takes 2.4 seconds to generate the public key using the fastest available standard machines. The large size fully homomorphic encryption system, that is, the one that is 2^15, which is 32768-dimensional lattice, requires two hours just to generate the key, which is 2.3GB in size.

So what we have is an entirely different way of encrypting and treating data. And it has almost none of the characteristics we're used to thinking about when we talk about standard deterministic crypto, whether it's symmetric or asymmetric encryption. Its process is just not suited for the way our current standard computers are structured, with 64-bit word lengths. These things need, actually this somewhat homomorphic encryption of the large size, 2^15, uses a 13 million bit integer to do its work. So we need a completely different technology of computing in order to work with this.

The feeling is, this has a huge future. We're at the beginning of it. And as cryptography always does, it will get faster. It will get better. People will come up with shortcuts. They'll come up with other ways to do things. This notion of noise in a lattice is only one of a number of nondeterministic encryption schemes that have been proposed. There are some others. There are some that use greatest common divisors. There are some that use large families of simultaneous equations where some statistical probability of individual equations being incorrect is, like, is a hard problem to solve. And so there are, like, very different approaches that are being explored.

I'm excited because this opens up something, like, completely new in the field of crypto research. And behind it will be tomorrow's applications that will work in a way which is completely foreign to the way we're used to thinking of things. And one can imagine, 20, 30 years from now, people will say, well, wasn't crypto always done this way? And it's like, uh, no. What we used to have was really stupid by comparison.

Leo: [Laughing] Big improvement.

Steve: Yes.

Leo: All right. I understood zero of what you said. But I trust that it all made perfect sense.

Steve: Well, the idea is that there is...

Leo: What's the executive summary for Leo?

Steve: [Laughing] It's that, if - let's see. If your data has noise, and you know how much noise it has, and you want to operate on it, as long as you don't do too much, then the noise doesn't overwhelm the signal. And then you can remove the noise. And that's useful because you can give the noisy data to somebody else, have them do the work, and, because it's noisy, they don't get any information from it. Yet the work they do is useful to you because you know how to extract the noise from the signal. Or, wait, the signal from the noise. So there's the even more condensed version.

Leo: Good [laughing].

Steve: And my goal here...

Leo: I'm trying to decide whether to pretend I understood that or just go, no, I...

Steve: My goal here, as with elliptic curve crypto, is not to turn us into homomorphic crypto people, but just so that - I'll bet you...

Leo: No, it's good.

Steve: ...a year from now it'll come around, and someone will say, oh, yeah, Google just added homomorphic encryption. It's like, oh, well, we know what that is. That's Podcast 376 on Halloween of 2012. Go back and listen to it. And you still won't know what it is.

Leo: Hey, did you have an opinion - we talked a lot about it during the week, and I think there's probably not much to say about the hack of, what was it, three million-plus Social Security numbers in South Carolina?

Steve: Yeah, I saw that.

Leo: Yeah, I mean, there's nothing much to say. They decided not to encrypt the data.

Steve: Yeah. It's unbelievable.

Leo: And the rationale was, well, you know, it's hard [laughing].

Steve: Yeah, well, it wasn't encrypted yesterday, and it was just fine.

Leo: Yeah. So we didn't need to...

Steve: And it's still not encrypted today, and we assume it'll be fine. Well, that assumption got broken.

Leo: And then, of course, they go on and on about how sophisticated the international hacker - I love the - the international hacker was who - amazing what the skills this person showed to get - to steal our data. Oh, well. There's nothing much to say.

Steve: Some secretary clicked on a link. I mean, that's what brought RSA down was some administrative assistant clicked a link in email, and that was the entre into the RSA network that allowed them to steal all the private keys.

Leo: Yeah. I guess the only silver lining is I would hope that CTOs in every other state are looking at South Carolina and going, god, I hope that doesn't happen to us. Maybe we'd better encrypt our data. But I feel so sorry for the three million South Carolinians who now their - now, some encryption was done apparently on the credit cards that they used to pay their taxes. But none on the Social Security number and personal information. So they're giving each and every one of them a year's free...

Steve: Credit report.

Leo: ...credit report [laughing].

Steve: Which is, you know, that's nice. But identity theft, boy, I'm sure you've heard the horror stories, Leo, about how difficult it is to get back your identity once it's been stolen.

Leo: Well, and the governor of North Carolina said that she, in fact, she and her husband have suffered identify fraud, and it was a very painful thing, and she knows how we feel. So that makes me feel better.

Steve: I wonder if they would have a philosophical objection to homomorphic encryption in South Carolina.

Leo: I don't know what it is. But, look, God did not intend for probabilistic encryption solutions. It's clearly written that deterministic solutions are the only approved stand.

Steve: And you know, Leo, if we had done this on April 1st, no one would have believed this podcast. They would have thought, Steve, that is the best darn...

Leo: What a fake.

Steve: ...April Fool's Security Now! podcast we have ever heard of.

Leo: Oh, lordy, lordy.

Steve: Right down to the name. They'd go, how did you come up with that?

Leo: Come on, you made that up, didn't you.

Steve: Oh, god.

Leo: Steve Gibson's at GRC.com. That's his website. That's where you can get SpinRite, the world's best hard drive maintenance and recovery utility. You can also get lots of free stuff there, all sorts of information, not the least of which is health information. If you go to GRC.com/health, you'll find links there to all the books and to our most recent Sugar Hill episode. We did a third episode on Sunday. That's also a TWiT Special, I think No. 143. But I'm sure you have a link on the website there, as well. And 16Kb versions of this show. It's for the people who are bandwidth-impaired. Transcripts, too, which makes it, for a show like this, kind of important. You can go over it line by line.

Steve: You can line your birdcage with it.

Leo: No. I think in fact this would be a great exercise for people who want to strengthen their brains.

Steve: It will be an exercise for Elaine, that's for sure.

Leo: Elaine's going, ohhhh. You can follow Steve on Twitter, @SGgrc. And, well, we do this show every - I won't be here next Wednesday. But normally every Wednesday, 11:00 a.m. Pacific, 2:00 p.m. Eastern time. That is going to be 1900 UTC next Wednesday because we go off summer time.

Steve: Yay, we fall back.

Leo: Yup, 1900 UTC. However, Tom Merritt will be hosting for the next few weeks as I head Down Under for a cruise, a geek cruise. So I won't see you till after Thanksgiving. Have a great Thanksgiving.

Steve: Will do. Yourself, too, my friend. Have a great trip.

Leo: See you in December.

Steve: And we'll talk to you in four weeks.

Leo: See you in December, Steve. See you in - is that weird, or what?

Steve: It is.

Leo: See you in December on Security Now!.