Crypto Series Wrap-up

Description: Steve and Leo conclude their multi-week coverage of the fundamental technologies underlying modern cryptographic systems. They discuss the number of 512-bit primes (two of which are used to form 1024-bit public keys) and the relative difficulty of performing prime factorizations at various bit lengths. They discuss the importance of, and solutions to, private key recovery using varying numbers of trustees; and conclude by explaining the need for, and the operation of, security certificates.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-037.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-037-lq.mp3

Leo Laporte: This is Security Now! with Steve Gibson, Episode 37 for April 27, 2006: Our Crypto Wrap-up. Bandwidth for Security Now! is provided by AOL Radio at AOL.com/podcasting. Security Now! is brought to you by Astaro Corporation, makers of the Astaro Security Gateway, on the web at Astaro.com.

Hello, Steve Gibson!

Steve Gibson: Leo. Good to be back.

Leo: Good – weird to see you. Actually, I'm sitting across from you. I never get to do it that way.

Steve: Yup, not so often now.

Leo: It's fun, though. I'm glad you're up here in Toronto. We're recording Call for Help and, as usual, taking a lunch break in the studio to do Security Now!.

Steve: Yup.

Leo: And we've been – it's been, I think, one of the top subjects we've covered. We've been covering crypto for the last four or five episodes, and it's been really great.

Steve: Well, I love it because these are technologies that we're all now using, literally in our

daily lives, on the web. And even though the deep math side is, you know, deep, the basic concepts, as we've been showing, are really easy to understand.

So it's funny, I got a really neat piece of email from someone who actually works at the CIA, who said people were looking at him funny when he was giggling, listening to the podcast – this is a few weeks ago when we were explaining about the Diffie-Hellman key exchange and how people could use exponentiation as the one-way function. It was easy to exponentiate; the reverse is to do an exact or a discrete logarithm. And what he loved was he had that ah-ha, like, you know, he got it. And he was giggling, walking through security at CIA Headquarters, because he was listening to Security Now!.

Leo: I don't know if you saw this, but apparently there was a mystery kind of solved on the grounds of the CIA this week. There is a sculpture that was kind of in the grounds of this headquarters at Langley, Virginia. And it's been there for some time. For 16 years people have been puzzling over the last part of it. Most of it, the first three – there are four sections. The first three sections had been deciphered in 1999. But ever since, the last 16 years, this fourth section has remained a mystery. Well, the sculptor announced, Jim Gillogly, that he – or actually Jim Gillogly had solved it. The sculptor's name is Jim Sanborn. He actually devised this with a retired chairman of the CIA crypto center, Edward Scheidt – announced, oh, I made a mistake, a booboo, there's an extra X.

Steve: Oh, my God.

Leo: So the puzzle, the "Kryptos" puzzle, I mean, there's websites devoted to it, everybody has been working on this for years...

Steve: So it had an error in it.

Leo: ...had an error in it. And it actually changes – he put it in for aesthetic reasons. It actually changes the message in the third section by just a few words. And now they're going back to work on the fourth section. It's actually kind of a neat little thing that has turned into a bit of a game at Langley, so that's kind of fun.

Steve: That's very cool.

Leo: Yeah. So there's a little crypto going on there.

Steve: Yeah.

Leo: So we were talking – we've talked about pretty much everything.

Steve: We've really covered...

Leo: Is there anything left?

Steve: Well, there are some interesting details and sort of loose ends that I want to talk about. You know, we started off talking about symmetric stream ciphers like RC4 used in WEP, and talked about the problems and the benefits of it. It's very fast. The problem is you can't ever use – you can't ever safely reuse the same key because we know what happens if you use a one-time pad more than once.

Leo: Right.

Steve: Then we talked about block ciphers and how much stronger they are, that they're a little slower than stream ciphers but still much faster than asymmetric crypto, which is what we then talked about, the whole public key technology. And then finally we talked most recently about cryptographic hashes, the idea of taking a document and distilling it down, whether, you know, whatever kind of message it is, distilling it down into a cryptographically strong, essentially sort of a token that is a signature or a fingerprint of that document. So those are sort of the foundations for a lot of this.

Now, people that have been following along have been writing in lots of stuff. And some of the questions that people have had have been common. So I wanted to talk a little bit, follow up a little bit on the asymmetric crypto side and talk about primes. Because there were people that suggested, well, okay, in terms of sort of like brute-forcing prime factorization, why couldn't you just have a table of all the primes? Of course, we were talking about the Sieve of Eratosthenes a couple weeks ago that is basically a table of primes. So I wanted to address this issue because a lot of people have wondered about it, just talking in terms of, like, how many primes there are. The sense that I think people have is that primes are rare.

Leo: It seems like they are. If you looked at a table, a ten-by-ten table, a hundred numbers, there were only, you know, a few primes in there. There are maybe 30 primes.

Steve: Exactly. And so I think that's what led people to believe, well, if there aren't that many of them, then why not just precompute them, do a table of them, and try them all?

Leo: Right. And in fact there are books of prime numbers that you can find these days.

Steve: Well, yes. They don't go up that high. The thing that we lose track of is that, again, and I've talked about this many times in the last few weeks, the notion of magnitudes. You know, we talk about 128 bits, 256 bits, 512 bits. But these magnitudes that this many bits represent are phenomenal. So a couple things. There's this Pierre de Fermat, the Fermat test, that back in the 1600s he showed how to find primes. And there's also something called the "prime number theorem" that talks about how many primes there are. Okay. Given 512 bits for a prime, there are fewer atoms in the known universe than there are prime numbers of 512 bits and fewer.

Leo: So there are quite a few.

Steve: Yes.

Leo: Wow.

Steve: So that's the point is that...

Leo: You can't make a book that big.

Steve: Exactly. No one is going to do a table. More atoms...

Leo: And even if you could, you couldn't search it fast enough or test it fast enough.

Steve: Well, exactly. More atoms in the universe – I mean, I'm sorry, there are more primes...

Leo: Than there are atoms.

Steve: ...512 bits or lower than there are atoms in the universe.

Leo: So that's why you use long numbers for this.

Steve: And it turns out it's not hard. We were talking about the computational burden of creating those. Now, when we talk about an RSA key that is 1024 bits long, what that actually refers to is the two prime numbers. When you multiple two 512-bit quantities, you get a 1024-bit quantity, that is, you sum the bit lengths when you multiple two binary numbers.

Leo: Well, that's interesting, okay.

Steve: So the 1024-bit asymmetric key...

Leo: Which is a pretty typical key size..

Steve: ...which is a very secure key size...

Leo: I use 4096. So, I mean...

Steve: Oooh.

Leo: ...you can get bigger.

Steve: Well, let's talk about this...

Leo: Let's stay with 1024.

Steve: Let's talk about this. First of all, what's really interesting is, you know how we talked about the Diffie-Hellman key exchange and how the one-way function there is exponentiation because it's easy to exponentiate; it's incredible difficult to reverse that and do a logarithm, that is, an exact or a discrete logarithm. Well, what's fun about the RSA public key technology is the one-way function is multiplication.

Leo: That's pretty simple.

Steve: It's just multiplication, exactly.

Leo: You're really good at that, yeah.

Steve: And the reason is that – so the reverse process...

Leo: Is factoring.

Steve: Well, it's factoring, but really it's just division.

Leo: Right.

Steve: You want to take this one large number. You need to determine what the two primes were.

Leo: And then you divide them.

Steve: And exactly. If you knew one, you would divide that by the total, and you've got the other.

Leo: The key is knowing the one.

Steve: Well, exactly, or knowing either one of them.

Leo: And now we know it's a 512-bit pool of data, so that's a lot of possibilities.

Steve: Well, it's a lot of possibilities. I did some research because I thought it would be fun to get some sense for the difficulty of doing prime factorizations right now. Okay. A recent record for factoring just 512 bits, that is, a 512-bit number that was composed of two primes – and this is just a couple years ago. So each of the primes was 256 bits long. A team used 292, so almost 300, off-the-shelf PCs, and it took them five months to do a prime factorization of 512 bits. Now, we know with a brute-force attack on symmetric keys, okay, where you have, like, 128 bits, a symmetric key, we know that every time you add a bit, the difficulty doubles because, if all you can do is brute force, you have to try them all, or try until you get lucky. But every bit you add doubles the number of possibilities because you had all the ones before with

that new bit off, and all the ones before with the new bit on.

Leo: That makes sense, okay.

Steve: So each bit doubles. It turns out that factorization doesn't double when you add bits, just because of the nature of the space that we're trying to explore. Every bit you add increases the difficulty or the length of time to find a prime factorization between 1.035 and 1.036. So not that much harder. Okay? 1.035 or...

Leo: Let's say it's one.

Steve: Somewhere between one – well, no. Well, no. It needs to be more than one.

Leo: Oh, yeah, one is not hard at all.

Steve: That wouldn't get any harder.

Leo: I get it.

Steve: So it is getting harder...

Leo: But just a little bit harder.

Steve: ...but not that quickly. It turns out, though, if you – now we know how hard it was to factor 512 bits. If you multiply that amount of hardness by 1.035 512 times, when you add another 512 bits, so now we're up to a 1024 bit...

Leo: Right.

Steve: ...now we're at somewhere between 3 and 30 million years.

Leo: Based on the fact that it took five months to do the 256.

Steve: Exactly.

Leo: Holy cow. Holy cow.

Steve: So we know very clearly how quickly it gets harder.

Leo: Right.

Steve: And we know how hard it is for the one we've done.

Leo: Right.

Steve: So we know that factoring a 1024 bit, which is sort of standard strength – yours is 4096. Now, knowing now how much harder it gets, you could compute how many millennia it would take.

Leo: Longer than the life of the universe remaining, I guarantee you. Yeah.

Steve: So people should just know that a 1024-bit number will take between 3 and 30 million years using state-of-the-art technology to factor.

Leo: What this underscores is, if it's going to be cracked, it's not going to be cracked by brute force. It's going to be cracked because somebody discovers a new way to find primes, there's some mathematical discovery that changes this. That's the only way.

Steve: And you know, in terms of difficulty, it's going to be cracked because somebody found the slip of paper you wrote it down on. I mean...

Leo: That's much more likely, frankly.

Steve: Exactly. I mean, it's like this technology just simply works.

Leo: Yeah. I mean, it's theoretically possible, though, isn't it, Steve, that some mathematician somewhere laboring in obscurity could – maybe even somebody at Fort Meade and at the NSA could have found a better way to factor primes that makes it infinitely easier.

Steve: That's why people that do crypto as a career, they are always careful with their language. They say "this is believed to be secure."

Leo: "Believed to be."

Steve: Yes. And they say...

Leo: That's the weasel word.

Steve: ..."as far as we know the difficulty is." But they say "as far as we know."

Leo: Right.

Steve: Because they always allow the possibility that something could be discovered.

Leo: But on the other hand, this is a branch of mathematics that is pretty well known, has been worked on since the time of Eratosthenes. It's not a new mysterious area.

Steve: Yes. It's, I mean, as we've said earlier, you know, grad students have cut their teeth in math trying to come up with new, fancy ways to do this.

Leo: Right.

Steve: Now, one thing we haven't talked about with public key technology is this notion of key recovery. That is, you create a key pair that's got a public key and a private key. We know that, because of the way this is typically used, you want to publish your public key. You want to put it out there so people can encrypt things to you, or so that you can encrypt things, and they can decrypt it using your public key...

Leo: Right.

Steve: ...knowing – proving that it was from you. Or, like, decrypt something that you signed to prove that it was you who signed it.

Leo: Right.

Steve: So in every case, the idea is that the private key is kept private, that is, you don't disclose it. What happens if you lose it? Because, I mean, that's a concern.

Leo: That's an avenue of attack.

Steve: People's hard drives.

Leo: Right, it's on my hard drive.

Steve: Well, if it's on your hard drive or...

Leo: It's on a USB key that I use to transport it around. I mean, it's all over the place.

Steve: That you might lose. Or your hard drive could crash. So it's not so much a matter of losing control of it as, say that for example a corporation had issued certificates – and we'll talk about certificates here later in this episode. We're going to talk about what a certificate is. But

the idea, they had issued public keys for every one of their employees.

Leo: Right.

Steve: And the employees were signing their documents. They were using these extensively. Now, an employee loses, for whatever reason, their hard drive crashes, they lose their token or something...

Leo: Hey, I'll put myself in that group. I've made over my lifetime about eight different keys, and I've lost all but the last couple. I can't revoke those old keys. That's gone.

Steve: Well. So imagine that a company wants to sort of store all of these keys somewhere safe.

Leo: Safe, yeah.

Steve: But remember like in the Cold War-era movies, or like the captain and the first mate on the submarine...

Leo: Strategic Air Command.

Steve: Yeah, Strategic Air Command, exactly.

Leo: You have two different keys, yeah.

Steve: They have, you know, around their neck, underneath their shirt is the lanyard...

Leo: And they both have to turn them at the same time.

Steve: Exactly. So the idea being you don't want to allow one person who might go crazy or, you know, or be compromised by a foreign government, to be able to launch the missiles.

Leo: Right.

Steve: You want to require two. Okay. So similarly, in a big organization where you've got lots of public keys, you want to somehow put in a database the private keys in case somebody truly does lose theirs, you want to do what's called "key recovery." How do we recover the key?

Leo: You know, I should point out that, if a bad guy gets my private key, he still needs to know a passphrase to be able to use it. So it isn't sufficient for him just to have the key.

Steve: Well, but the concern is not that you lost control of it. The concern is that you...

Leo: I lost it.

Steve: You lost it.

Leo: Right. But I'm just pointing out that losing control of it isn't necessarily the end of the world...

Steve: Correct.

Leo: ...because you need a passphrase to use it.

Steve: But losing it completely is the...

Leo: Well, it's bad news.

Steve: ...is the end of the world.

Leo: And I have, in fact, lost my...

Steve: Yes.

Leo: You could send me messages I can't decrypt because I no longer have the private key.

Steve: Or you're no longer able to sign your messages because you use your private key for signing messages.

Leo: Right, right.

Steve: So we need a way to allow those keys to be recovered.

Leo: Yes.

Steve: Now, say that we took the private key and chose a big random number. Well, chose a symmetric key size random number, which we know is, like, 128 bits. So we choose a random number. We symmetrically encrypt the private key with that, and we store it in our big database. Okay, well, we've only kind of shifted the problem because now we have a symmetric key...

Leo: We can't lose.

Steve: ...which we can't lose...

Leo: Right.

Steve: ...which we need to use in order to decrypt...

Leo: Right.

Steve: ...the private key that we're trying to store. Now, okay. Say that we had two people, and we needed them both. Well, you could take the 128 bits...

Leo: Split it in half.

Steve: ...and split it in half...

Leo: That's good.

Steve: ...and give them each 64 bits.

Leo: Yeah. I need you and you need me. We can't decrypt without it.

Steve: Exactly. The problem is, now each of them have 64 bits of the key, that is, now they only need another 64 bits.

Leo: Oh.

Steve: And that's no longer strong.

Leo: Right.

Steve: Because we know that 64 bits can be brute-force attacked.

Leo: Right, right.

Steve: Not soon, but it's still possible. It gets worse. Say that, okay, now we have another problem. Say that we've split the key in half, and one of the people we need is on vacation. We can't get a hold of them. They're in Tahiti.

Leo: Right.

Steve: Okay. So now what we want is we want a more sophisticated system where, say for example, we need two out of three people who can be present in order to decrypt our database in order to get back the keys. Well, that would require that we take our 128 bits and split it into three pieces and give each of the three people two different chunks of three. Okay? So, like, the first person gets pieces one and two; the second person gets pieces two and three; the third person gets pieces one and three. Now, that means any pair of them have all the bits. But now we've made that prior problem even worse.

Leo: Because now it's only, like, 43 bits, and it's really insecure.

Steve: Exactly.

Leo: Now you actually are getting to a level where it's been cracked. 40 bits has been cracked.

Steve: Yes. So we've got to scrap this whole idea.

Leo: Bad idea.

Steve: Now, here is a concept, Leo. This is one of those, I mean, our friend who works at the CIA is going to be giggling as he goes through security. This is just a toe-curling concept.

Leo: Okay.

Steve: Okay. So imagine a sys- we want a system where any number of people can be given information such that any group of them that we decide are necessary to – almost like voting. They all – they get – and we would call them our "key trustees."

Leo: Yes.

Steve: So we're going to make them the trustees of the database. We don't want to allow any few of them to be able to access the data. We need a majority of some sort...

Leo: A quorum, okay.

Steve: ...in order to access it. Okay. Let's just take the case of two people . We already saw that we can't just give them each half the key.

Leo: Yeah, that's not going to work, right.

Steve: Because that's no longer secure. Okay. We know from geometry that...

Leo: Uh-oh. That's when I go south.

Steve: No no no, this is good.

Leo: We know from geom- okay.

Steve: And you're going to be able to visualize this.

Leo: I'm closing my eyes now.

Steve: Okay.

Leo: Okay.

Steve: We know from geometry that a line is defined by two points.

Leo: Yes.

Steve: That is, any two points, and you draw a straight line through it. Okay. Now, imagine we have an x,y grid of Cartesian coordinates, and the line – we just draw a line. It crosses the y-axis at the key. That is, the value where it crosses is the key.

Leo: Right, got it.

Steve: Okay. Now, if a single point is on the grid, we know that there's an infinite number of lines.

Leo: That can just rotate around that point.

Steve: It could rotate around that point. Two points define the line.

Leo: That's one and only.

Steve: And we choose – so we draw a line that crosses through the y-axis at the key.

Leo: Yeah.

Steve: Now we assign as the secret points on the line, so that two people each get a point.

Leo: But nobody has two points.

Steve: Nobody has two points.

Leo: So nobody can decrypt it.

Steve: And what's cool is you could assign as many different people points on that line...

Leo: It's an infinite line.

Steve: ...as you want to. And so what this means is any two – so you...

Leo: But the point is a pair; right? It's...

Steve: A point is a pair.

Leo: ...an x and a y.

Steve: So an x and a y coordinate. But that, the coordinate by itself, provides no information about the key.

Leo: Because it also has an infinite number of possible crossings of the y-axis.

Steve: Exactly. One point, and then there's...

Leo: Tells you nothing.

Steve: ...an infinite number of lines. But any two points, now you can compute...

Leo: We draw a line to the y-axis.

Steve: You could draw a line...

Leo: Wow.

Steve: ...and get the key. Isn't that neat?

Leo: That’s really slick.

Steve: Okay, so now, say that you wanted...

Leo: So we give out a bunch of these points.

Steve: So you just give out points on the line.

Leo: And any two people...

Steve: Any two people who’ve received...

Leo: ...can give you the key.

Steve: ...points are able to recover it.

Leo: Oh, that is so cool.

Steve: Now, say we want to make it a little more fancy. We want to require three people.

Leo: Yeah, how do you do that? Slope.

Steve: Three points define a parabola. ‘Cause, think about it. If you have a curved line, as we know, the two points is now a family of parabolas...

Leo: Yeah.

Steve: ...but a third point nails the parabola.

Leo: It’s a unique parabola.

Steve: Three points define a parabola. So all we do is we create a parabola that passes through...

Leo: The y is the...

Steve: ...the y axis.

Leo: ...peak or the trough of the parabola.

Steve: It just passes through. And then all we need to do is pick points. We can assign as many points as we want on the parabola, giving those coordinates to people. And three of them have to get together...

Leo: Perfect.

Steve: ...in order to recover the key.

Leo: So here's our stumper for the week.

Steve: Isn't that cool?

Leo: What if you want four people to have a key?

Steve: Well, it turns out that this is completely generalizable. A parabola is a curve of second order. And you are able to do nth...

Leo: An infinite number of curves.

Steve: ...nth order curves. So it's simple polynomial math.

Leo: I love it.

Steve: And it allows people – it allows you to distribute the knowledge such that you can completely define your policy – how many people have points, how many of those people have to share their points in order to unlock a key.

Leo: And so this is how key recovery works.

Steve: This is one of the predominant key recovery techniques.

Leo: What a cool idea.

Steve: Isn't it? I just love that.

Leo: That is really slick.

Steve: Well, the final thing to talk about in this whole series is certificates. We've glanced on certificates. We've talked about web server certificates and in PGP land. We've talked about personal certificates. Someone did write saying that Thawte is still making available free personal certificates.

Leo: Yeah. They reduced the amount of time you can use it for and so forth.

Steve: Ah.

Leo: They were purchased by VeriSign...

Steve: Right.

Leo: ...and of course they've changed. Thawte used to be really almost a nonprofit. They really were supporting free...

Steve: Right.

Leo: ...email certificates. There's sort of, you know, there's pros and cons to the way PGP does it and the way certificates do it. And I have both, but I like PGP. I do.

Steve: One thing I didn't realize, but again I've been sort of extending my reach recently, and that is that S/MIME, Secure MIME, is there, and it's in all of our email readers now.

Leo: Yes. And PGP uses S/MIME.

Steve: Okay.

Leo: It can use S/MIME. It doesn't...

Steve: Well, because it is a...

Leo: So what is S/MIME?

Steve: S/MIME, MIME stands for...

Leo: Multipurpose...

Steve: Multipurpose Internet Mail Exchange.

Leo: Right.

Steve: Which is a well-defined format for creating non-textual content in messages, you know. So a plain...

Leo: Binary files...

Steve: Exactly. If you have an attachment, that's actually a so-called MIME type that is defined, and there is an encoding that moves the binary into text so that it's able to move through the email system. Anyway, we're going to do an entire podcast on S/MIME in the future.

Leo: Oh, good. Because I've wondered, you know, that's an option in my PGP, use S/MIME, and I've never known why or should I.

Steve: And it's in everyone's copy of Outlook Express.

Leo: Built in.

Steve: And so you're able to get a certificate and add that certificate to Windows and then turn on S/MIME. And in fact you can add the public certificates of people you correspond with, and your client will automatically encrypt the mail if you're sending it to someone whose certificate you have.

Leo: Very cool.

Steve: So it's very cool. We're going to explore it in detail. But what is all this certificate? Okay. The problem with public keys is that they're public. I mean, the benefit is they're public. But what if somebody claimed that, you know, what if Joe claimed that Alice's public key was his?

Leo: Right.

Steve: I mean, he could make the claim because Alice published her key. I mean, the value of it is that it's public.

Leo: Or worse, Joe could create a key that says he's Alice, and you'd have no way really of knowing that it wasn't Alice's key.

Steve: Well, yes, that's a very good point. The problem is that keys themselves have no identity.

Leo: Right.

Steve: Keys are just a blob of bits. As we know, they're...

Leo: I can assert that it's mine, but you have no way of proving it.

Steve: Exactly. So here is what a certificate is, and I love it for its simplicity. A certificate is nothing but the binding – and we'll describe what that means – the binding of identity information to a public key. That's what it is.

Leo: And using a trusted third party.

Steve: Well, that's where this binding comes from. But even stepping back from that, because you don't need a trusted third party to have a certificate.

Leo: No, I can make my own certificate. But it has the same issue. I can make my own certificate.

Steve: Right.

Leo: It's not really a binding.

Steve: So what a certificate is, it's the binding or the association of identity information of some sort.

Leo: Right.

Steve: You know, your name, your address, your company, whatever it is, to a public key...

Leo: Yes.

Steve: ...to give the public key some identity. Now, the veracity of that binding, that is, what people consider that information to be worth, is who is standing behind the representation.

Leo: So, for instance, in OS X and many other operating systems, I can create a public key, I can create a certificate, both of which have really no authentication because I'm asserting that they're mine, but anybody else could do that. You could create one that says that you're me. There'd be the same value.

Steve: Exactly.

Leo: So that's where the trusted third-party...

Steve: So in order to give a certificate value, you need somebody to vouch for it.

Leo: Yeah.

Steve: You need somebody to say, I know this person, this person has proven that they're who they say they are, and this is their public key.

Leo: It's kind of like a notary public. When you sign a document and you go to a notary, this notary is entrusted by the state to look for your driver's license or whatever and...

Steve: To do due diligence.

Leo: They have to do the due diligence.

Steve: Right.

Leo: They have a stamp and so forth.

Steve: So here's the process. An individual produces a key, a public key pair that of course as we know has a public key and a private key, and they produce their own identity information and whatever they need to do to prove that this identity information is valid within the constraints of their application. They sign what's called a "certificate request" with their private key. Now, we know that that can only be verified with their public key. So they sign the certificate request with their private key, never giving it out, but they sign it. The reason they do that is so that the so-called "certificate authority" can verify that the public key they're providing is matched with the private key they used to sign the certificate. Therefore they are proving when they sign this that they actually own the private key...

Leo: Both sides of the key pair.

Steve: Both sides, exactly. So the certificate authority takes the identity information and this user's public key; creates a message, like we've talked about before; makes a hash, using hashing; and then the certificate authority signs that with their private key. So basically you're having a third party sign your assertion that this is your public key and your identity.

Leo: That's the equivalent of the notary's stamp saying, yes, this matches.

Steve: Right. Now, the next part of this is – so, okay. What we've done is we've created a certificate. That is the certificate. Now, to verify that, we need – since the certificate was signed with the certificate authority's private key, we need their public key in order to check it. Well, our browsers and our operating systems come with, like, 30 or 40 public keys with long expirations, like 2020, 2040, I mean, way in the future. So they're saying this certificate is valid

through this length of time, and it contains the public key of the signing authority who verified.

Leo: And these are people like VeriSign and Thawte.

Steve: And Thawte. Even the U.S. Postal Service has a certificate that is available...

Leo: See, I'd trust them.

Steve: Yes.

Leo: I know they're going to be around for a while.

Steve: Yes. And so that's really the way this functions.

Leo: Now, I should mention that, as complicated as that process sounds, it's actually automated by most email programs. It's pretty straightforward. They send you an email, and you get it back, and...

Steve: Exactly. And in fact a dialogue box pops up, you...

Leo: ...accept it.

Steve: ...you fill out the form, and, you know, the public key gets made, and it all happens just easily.

Leo: The free ones, generally the only thing they verify is the return email, is your email address, because they send it to your email address. The paid ones may get more information from you.

Steve: Well, and for example, as someone who has a bunch of web server SSL certificates...

Leo: Same idea; right? A server certificate's the same as an email certificate.

Steve: Because, exactly, someone's connecting up to GRC over a secure connection. And somebody is asserting that this is really GRC they're connected to. And so when I get my certificates from VeriSign, they look me up in Dun & Bradstreet. There's a fax and a phone call and an email loop. They go through some trouble to verify that this is really who I'm claiming it is.

Leo: Right, right. Now, PGP, which doesn't use a certificate system, has a little kind of less reliable system called the "web of trust".

Steve: The web of trust; right.

Leo: And if you use a PGP key, the first time you use it it's kind of untrusted. And what you hope is that people will, as they get your key and know, oh, yes, I know this was an email from Leo, will go to the server and sign your key and say, yes, this is Leo's key. And eventually your key will be signed by enough people that the level of trust will be quite high.

Steve: And that's a cool system. I mean, it's sort of the way Google works by, you know, ranking sites based on the ranking of sites that link to you.

Leo: It's a recursive system, but it works.

Steve: They're sort of a, yeah, there's a web of link quality endorsement as opposed to, like, a web of trust.

Leo: In fact, as we mentioned, people often have PGP key signing parties, where they all get together and sign each other's keys.

Steve: Right.

Leo: Which I think is a great idea.

Steve: In order to create an immediate veracity behind the key.

Leo: Right. Somebody suggested – because I'm actually using a new key because I lost some of my old keys, and I don't think it's signed yet. And somebody suggested, oh, well, you should do a podcast key signing party. I haven't figured out quite the logistics of how we would do that. I don't think we could.

Steve: Yeah, yeah.

Leo: I was thinking maybe I could give out a code word on the podcast, and they could send me stuff. But I haven't figured that out yet. I do get, now, thanks to this, though, a lot of encrypted mail because people know I use PGP.

Steve: Right.

Leo: And I have quite a few email buddies. And I'm sure the NSA is looking carefully at all of this encrypted traffic.

Steve: What's all this encrypted traffic coming and going to and from Leo?

Leo: Well, and that's one of the arguments for more people using PGP and other encryption technologies is because, right now if you use it, it's a red flag.

Steve: Then you don't stand out; right.

Leo: But if everybody's using it, then it's just – it's like using an envelope in your mail. You know, if everybody sent postcards, and suddenly you started sending mail with envelopes...

Steve: Oh, what's he trying to hide from us; right.

Leo: So let's all please start using envelopes, if you don't mind.

Steve: So that really wraps up all of the fundamental technology of crypto. There are some cool things we will be talking about in the future, although we're going to change topics now and go on in different directions. But we will in the future talk about how SSL works in detail, how PGP works, how Secure MIME works. And, you know, other things, because after all we're here talking about security. Now with this set of fundamental tools we can talk about symmetric keys and private keys and hashes, and people will know where we're coming from.

Leo: Isn't that great. Isn't that great. Steve's, of course, the guy behind SpinRite, a fantastic program for disk maintenance and recovery. We encourage you to visit GRC.com and get your copy of SpinRite. If you've got a hard drive, you need SpinRite. And of course show notes for Security Now! are available at GRC.com/securitynow.htm.

Steve: And transcripts.

Leo: Thanks to Elaine, we've got transcripts, and 16KB versions for the bandwidth impaired, all at GRC.com/securitynow.htm. We also invite you to visit our fine sponsors, Astaro, makers of the Astaro Internet Gateway, their great Astaro security software. I just got my 120.

Steve: Oh, cool.

Leo: And I'm setting it up, and it is – I just feel – it just gives you this sense of security that's just fantastic.

Steve: Well, because it is a managed system; right?

Leo: Right, yeah.

Steve: Yeah. And so...

Leo: So it's a router, but it's a router running open source software with – and, you know, these are by subscription, but you can get anti-virus, you get anti-spam, and it's really great.

Steve: And it's being updated transparently as their database grows.

Leo: You could try a free version. In fact, there's a home version that's absolutely free at Astaro.com. But I might mention, somebody pointed this out, that you can get the free version, put it on an old box. But for 79 euros a year, subscribe and you get ClamAV, you get two anti-viruses, anti-spam, automatically updated. I mean, frankly, compared with the other anti-virus/anti-spam solutions, that is very economical. That is probably the way to go if you've got an old PC.

Steve: And it protects your entire network.

Leo: Your whole network.

Steve: Without needing something then on every one of your machines.

Leo: Yeah. Stateful inspection firewall, I mean, this is topnotch security. Astaro.com. So I guess we'll be back next week with more fun security information?

Steve: I'm sure we will. We've got lots more to do.

Leo: It's great to see you in Toronto.

Steve: Likewise, Leo.

Leo: All right. And we'll see you all next week, next Thursday, for Security Now!.