Achieving Internet Anonymity

Description: Last week Steve and Leo discussed the social implications and the social power of Internet Anonymity. This week they discuss the technology of Freenet and TOR (Onion Router) networks, and Steve describes the detailed technical operation of both systems.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-070.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-070-lq.mp3

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

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

This is Security Now! with Steve Gibson, Episode 70 for December 14, 2006: Freenet and TOR.

Security Now! is brought to you by Astaro, makers of the Astaro Security Gateway, on the web at www.astaro.com. And by Dell. For this week's specials, visit TWiT.tv/dell.

Time for Security Now! with Steve Gibson, the security guru from GRC.com. And we are so glad to get you back on every Thursday, Steve. We have not missed an episode.

Steve Gibson: We're never going to, Leo. We recorded last week's actually yesterday, which was three days early for last week's episode, and now we're recording this one on the following day, on Tuesday...

Leo: A week early.

Steve: A week and two days early because you and I are going to be together in Toronto next week, so.

Leo: But I appreciate your doing that, and your willingness. While much of the TWiT network is going to go to sleep for the two weeks of Christmas and New Year's, you continue on. There'll always be a Security Now! if you're getting desperate.

Steve: And you know, Leo, I'm very competitive, and I'm watching the fact that our episode

numbers are creeping up on TWiT because every time TWiT takes a pause or a little hiatus, it's like, oh, good, I get a couple numbers closer.

Leo: You are going to beat them. Let's see, you're at 70. TWiT's at 79, 80. Oh, it won't take much longer. I think we'll take a week or so off because I'm going to Mexico for Christmas. So I think we'll take a week or so off for TWiT.

Steve: Well, that'll only lose you one or two episodes.

Leo: Yeah. I'll lose a few more just for you.

Steve: Okay.

Leo: So last week we talked a little bit about anonymity on the Internet, how difficult it is to achieve, first of all, but also the political ramifications. And, you know, there are pros and cons. But I think we concluded ultimately that if you believe in free speech, you have to support anonymity of some kind.

Steve: Right. I felt a little concerned that last week's episode might have been a bit of a fluff episode, that is, you know, not deep security technology. We're going to make up for that today. But I really wanted just to take the time to talk about the issue, like the social side of this, because, you know, it is important, and we're talking about very, very significant technologies, that powerful entities like the RIAA and the government with the DMCA copyright stuff and foreign governments that want to regulate free speech don't take this issue of anonymity and privacy with a grain of salt. They've got organizations that are all about preventing people from being able to speak freely. And as we know, our own U.S. government has now disclosed that they're recording and wiretapping as much of our country's telecommunications as they're able to.

Leo: And from a nonpolitical point of view, we just learned that the government is going to start requiring businesses to save all emails and instant messages for three years. So even at work you don't have any privacy, especially at work perhaps, not even "even." So I think it's important that we did talk about that, mainly because we often in this business talk about the technology, but don't talk so much about the ramifications. I know when I sit down with Phil Zimmerman of PGP, and we did a number of times on the Screen Savers, I always try to bring up this issue. And he's good about talking about it. But frankly, as most engineers are, he'd rather talk about the technology. So I think, yes, it was fluffy, but we needed to do it.

Steve: Well, that's behind us now.

Leo: So get ready.

Steve: Exactly. And in fact we're going to be using a lot of the foundation that we established, I guess it was either early this year or late last year, all of that crypto stuff. We spent a number of episodes talking about public key, asymmetric, and also so-called private key, or symmetric cryptography. We're going to need that foundation. And remember at the time that I wanted to lay that foundation so we would be able to have then conversations that assumed people

understood that stuff so that we could talk about all the fascinating ways that this kind of crypto is able to be assembled and put together in different – well, essentially to achieve different sorts of goals.

Leo: We're going to get to our subject in just a moment. But first a short commercial break for Astaro and Dell. As always, this episode of Security Now! is brought to you by the good folks at Astaro Corporation, makers of the Astaro Security Gateway. If your small or medium business network needs a superior device to protect you from spam, from viruses, from hackers, as well as complete VPN capabilities and intrusion protection and content filtering and an industrial-strength firewall, all in a very easy-to-use, high-performance appliance, you want to check out the Astaro Security Gateway, Astaro.com. Or you can call 877-4AS-TARO and schedule a free trial of an Astaro Security Gateway appliance in your business, really see what it can do for you. They know once you try it out you're going to want one. And by the way, non-business users, you can use their software absolutely free. It's open source and available at Astaro.com. We thank them for their support of Security Now!.

Also we want to thank Dell for supporting us. Dell's been advertising for the last three months with great success, and we really thank the TWiT audience, the TWiT Army, for making it so. We've used Dell for so long on the TV shows, and I own Dell computers. In fact, my new laptop's a Dell. What I've done is put some links to the newest Dells we just got, both for the TV show and me personally, on our Leo's Picks Page. That's TWiT.tv/dell. Just go there, visit, take a look, click the links. If you want to buy something else, let's say one of those great Dell 24" monitors – I love those, Steve has a bunch of them – just click the link at the top there that says "Go to Dell." And anything you buy after that point is credited to TWiT, so we do appreciate that. That's TWiT.tv/dell.

And now back to Steve Gibson and Security Now!. Is crypto always the foundation of anonymity? It has to be scrambled, I guess, to be anonymous.

Steve: Well, I guess. I mean, if you look at a real world example, where somebody just writes something on a piece of paper, if they wrote it with the other hand than they normally use, and they passed it through three or four people, none of whom knew the person before the one that they'd received it from, and so no one knew who they were going to be giving it to, I mean, you could invent real-world anonymizing solutions which are non-crypto oriented. So I would say that in general crypto is not necessary. But it turns out that in order to make it, certainly on the Internet, in order to achieve the goals of not being able to associate the sender and the receiver of information, crypto really does play a strong role because it allows you to have provable enforceability of these things.

So let's step back a little bit and just notice that one of the reasons anonymity is so difficult – and one of the first things I said last week was Internet anonymity is much harder to achieve than most people appreciate. The reason Internet anonymity is so difficult is essentially, like so many things the Internet was not designed to provide, it was never designed to provide anonymity to people. That just wasn't in the design specs. It wasn't part of the plan. And the most perfect example of that fact is that IP addresses are the way packets route from one end to the other. So as we know, all Internet endpoint systems – machines, routers, whatever – are identified uniquely on the 'Net by an IP address. Well, I mean, technically it's anonymous inasmuch as it doesn't tell you the name of the company or the name of the person with the IP. But it is a tag that can be used to uniquely identify every sender and receiver of traffic on the 'Net.

Leo: So it's the first thing you want to get rid of in anonymity.

Steve: Yes, exactly. It is the way the 'Net is structured and the way data moves is there's never been a notion on the 'Net of the Internet being about anonymity. So the point is, anything we do to add that is a layer on top of the existing structure, which was just never designed to be anonymous. So it's a hard thing to do.

Another example from things we've discussed before is SMTP headers. We've been talking about, for example, if I send email through Google but from my own system's email client, rather than from a web page at the Google server, there are headers which are added that identify my IP as the originating IP. And in fact, that's the case even if you use Hotmail or Yahoo! and you use their browser interface. The IP you sent from is there. And in fact the SMTP protocol, the protocol for routing email across the Internet, in order to avoid the potential for routing loops, that is, where email is sent from one SMTP server to another, it sends it to a third, and due to some misconfiguration, the third server sends it back to the first. Well, if there wasn't some way to record the path the email had taken so far, then it would be possible for the email to get stuck forever being forwarded through multiple SMTP servers in some sort of a routing loop.

Leo: So the very structure of the Internet requires real identification.

Steve: Exactly, exactly. Many things about the 'Net are assuming that you're going to create some sort of a record of the path that the data took or certainly the IP that originated the data so that you're able to send something back. So as a consequence, all of the technology we've got in place is about being non-anonymous. Not necessarily identifiable to an individual or a corporation, but certainly back-trackable to the physical, sort of the electrical source of the data.

So two technologies we talked about last week are different in what they achieve. I want to talk about Freenet only briefly because there isn't that much to say about it. And then we'll sort of finish out by talking in detail about how TOR works.

As I mentioned also last week, many anonymizing systems deliberately don't allow people to anonymously create content. That's not what they're about, you know, the famous Anonymizer.com system which has been around for a decade, they're about allowing users of their service to anonymously visit websites. And so those users are pulling existing public content, but their own identity, their IP address, and presumably cookies and other bits of debris that might be part of the HTTP transaction, those are being deliberately cleansed and filtered by this anonymizing proxy, and their actual address is being hidden.

We remember also that, by default, a proxy will include the IP of the user for whom it is performing the proxy service. For example, cable modem companies use a caching proxy in order to minimize their use of bandwidth for all their consumers who are, for example, going to the same page or wanting to download the same GIFs and so forth from web servers. If the caching proxy has the data, it doesn't need to ask the server for it again. That minimizes the ISP's transit bandwidth and allows the ISP to keep the bandwidth within its own organization, for which it pays a much lower cost because it's using its own infrastructure. But when requests leave that ISP, almost always there will be a tag in there saying, I am making this request on behalf of the following IP. So again, that proxy is not helping anonymity at all. And it's very possible, then, for the website to see where the request really came from.

So an anonymizing proxy, that is, a proxy that is deliberately designed to hide the identify of someone for whom it is making the request, it will not add sort of those optional headers, which really don't have to be there anyway, and it will make the request just as if it was making it on its own behalf, and then turn around and send the response back. So there's anonymity being provided there. But as we also mentioned last week, it's not very good because it really wasn't meant to be ultra robust.

Leo: Well, doesn't the server know where it came from? And that's part of the problem. The server can keep track of that.

Steve: Well, yes. Certainly their own records – if they were maintaining records they could. But imagine an external agency, for example, a government that wants to know who is making...

Leo: Who's doing bad stuff.

Steve: Yeah, exactly. Who is using an anonymizing proxy in order to hide their identity? All that person would have to do is analyze the traffic patterns coming to and from. Well, okay. In the simplest case, imagine that only one user was using the proxy. Well, if one user was using the proxy, packets would go to the proxy, they would come out the other end, go to a web server, the responses would come back from the web server, and they would go back out of the other end of the proxy, back to the user. So obviously anyone looking at the traffic could determine which user was visiting. They would know the IP...

Leo: But that's the trivial case. There's a lot more than one person using these things.

Steve: I wanted to point out, though, that that's the level of, I mean, that's the way this can be busted is not only using timing, but using packet sizing. Because another way to associate incoming and outgoing traffic is to look at the packet sizes which, you know, may or may not be changing dramatically in size. So my point is that, yes, when many users are using a single focal point, then it's more difficult. But more difficult still being way, way short of being impossible by doing traffic analysis. So traffic analysis is something that the people who built the TOR network, for example, and to a somewhat lesser degree the Freenet network, that's something they recognized as an Achilles heel for anonymity busting. And they've gone to substantial lengths to get around that. And that's what we'll be talking about when we talk about TOR.

But Freenet differs from many of these other anonymizing systems because it is all about storage. That's what it's about. It's about basically creating dark networks, as they call them, of storage. Users of Freenet give up – they basically share unused hard drive space to participate in this Freenet database, sort of in the same way that the people who run the SETI screensaver are contributing their unused CPU cycles, and sort of the way that users of traditional peer-to-peer filesharing like Kazaa and Limewire and all those, they're basically allowing their unused bandwidth and also their storage in some cases to be used.

Similarly, Freenet users are saying, okay, I want to participate in this distributed database, so I'm going to give up a chunk of my hard drive in return for being able to use chunks of everybody else's hard drive in this network. The way the system works is – and probably since so many people are probably familiar, at least in passing, with Kazaa, for example, or any of the traditional peer-to-peer filesharing systems, those systems have a number of problems. First of all, and probably mostly, the data being sent back and forth is not powerfully encrypted, so it's possible for someone to create a sort of a pseudo user and obtain known copyrighted files from somebody else. And you know who has them. The files stored on people's hard drives are not encrypted, they're just the regular file. And they bear their regular filename. So it's very easy for the RIAA to issue subpoenas, cause the police or FBI or whatever law enforcement to visit someone's home, grab their computer, do a directory listing, and see all of the copyrighted material these people have.

So what Freenet has done is they have completely solved all of that problem. That is, as we were saying last week, it's a fact that you cannot have enforceable copyright if you have

freedom of speech. And I want to make very clear that I'm not advocating any breaking of copyright. I'm just talking about the technology. We covered the social and political aspects last week. So I'm just wanting to explain how this works rather than running around cheering it on. That's not what I'm here for. What the system does is, someone who wants to submit a file, whatever kind of file, could be political agenda, it could be anything, I mean, Freenet basically is a very secure, anonymizing, distributed database. Somebody submitting the file generates a hash of whatever they want to call it. So they could call it Batman_Begins_DVD.iso, I mean, that's what they can call it if they want to. But it is hashed using 160-bit SHA1 hashing.

And we know from prior Security Now! episodes where we've talked about that, what that does is a cryptographic hash takes a string of plain text, that is to say, the name of the file, and converts it to a very likely to be unique hash, that is, 160-bit, that looks just like random data. If you change any character in the name, the hash ends up having, if it's a good hash, about half of its bits will get flipped, and it'll just look like it's a random flipping of bits. There's no way – and this is the key – there's no way to go backwards, that is, you put the filename, you know, Batman_Begins_DVD.iso, into this hashing function, out the other end comes 160 bits.

Well, every time you put the same name in, you're going to get the same 160 bits out. But there is no way of having the 160 bits to reverse that process. You can do what's called a "dictionary attack," that is to say, you could guess that something was this name and put it in and see if the 160 bits matched. And if you got a collision, there's a high probability that the same string was used, but it's not provable. That is, even then, even when you get what's called a "hash collision," meaning that you put something in and you got the same result out, there's never a way to prove that there – well, because in fact mathematically we know there are a great many strings in theory that can generate the same 160 bits. So you have provable deniability there.

Well, through this process of encryption and hashing of descriptive keys, Freenet is able to provide anonymous storage of anything people want to store. There's a problem with what's called a "flat name space," meaning that all people who were using Freenet were using sort of like the same hash. So what they've done is they've created this notion of private name spaces, meaning that an individual user will have their own handle, and so their own handle is used along with the file name or file description that they want to submit to create a hash. And that prevents somebody else with their own handle who submits a file that happens to collide in terms of hashing, it prevents that collision from happening because individual users have distinct handles.

There's also, I mean, it's such a complex protocol. I've looked at it. I've understood what they're doing. But it's really not necessary to get into the minutiae of the details of how they perform encryption. But I do have a link on our show notes for a PDF that completely describes this, if this description ends up interesting someone who wants to really get into it, because these guys have worked out the system such that a Freenet system of these nodes, all nodes are peers. They're identical. They don't have knowledge of each other beyond their own local neighbors. And that's a deliberate measure to increase anonymity.

So the idea is, when someone inserts a document, a file of any type and description, into the system, it just gets absorbed by this distributed network. There are indexes running on nodes that index based on the proximity of the hash. So the system self-organizes. And it means that, if you were to tell somebody else, go get this file from Freenet, they are able, from no matter where they are geographically, to – into their own node to say, I would like to retrieve this file. They do it using the plain text strings they were given, meaning the user's name, or the user handle and the file description. The Freenet hashes that into these indexes. From that and from other material, they're able to generate cryptographic keys which are able to decrypt the file because all the contents is stored encrypted in a way that is really secure. So there isn't a way, if someone did bust down someone's door and say, hey, we know you're running a Freenet node, we want what's on your drive. Well, all that's there is a bunch of random bits.

Leo: I have to say, I mean, this all sounds really good. And I guess we talked about this last episode. I just want to kind of underscore this because I know people are thinking this as they listen. Okay, so I'm allowing people to store pirated movies on my hard drive? Why is that a good thing?

Steve: And now, Leo, now that I've sort of better explained the power of this, now you can understand why I wanted to spend some time last week and say, well, you know – and this is also why I just said I'm not advocating this.

Leo: I bet there's a lot more pirated stuff, or maybe illegal stuff, than there is dissent and, you know, documents from whistleblowers and things like that.

Steve: I think that's probably the case.

Leo: Like 99.99 percent.

Steve: It is a powerful archiving system for trading and posting anything. And again, as I said last week, and this comes from the Freenet papers, if you have – in order to get free speech, you must have anonymity because speech is not going to be truly free if it's not anonymous, or if you at least don't have the option of it being anonymous. And similarly, copyright requires enforcement, but enforcement requires a lack of anonymity. Otherwise there's no one to enforce against. Which means that, if you're going to have free speech, you are not going to have copyright enforcement. And so...

Leo: Ah ha. You did say that last week. Now I understand why.

Steve: Right. And so the point is, these Freenet guys, the people who have designed this and put it together, they're adamant about the necessity for free speech. Free speech means anonymity. Anonymity means non-enforceability of copyrights. And there just isn't a way around it. It is a legal, social, political dilemma. But our society has lots of those. And so here's one more. But I'm glad you reinforced the idea that what this really means is people who participate in Freenet who want the option of accessing this data and who are contributing a chunk of their hard drive, their hard drive is probably storing really bad stuff. Because just in general, bad stuff is the stuff that has to hide. And this is all about hiding. You know, it's about anonymity and hiding. Okay. So...

Leo: After we talked about it last time, I went and I looked at the site, and I thought, I'll download this, install it, because I want to support freedom. But I decided not to for that very reason. I thought, but I don't know if I want to support storing other stuff. It's a very tough choice.

Steve: And as we talked about anonymity, even anonymity by itself is a double-edged sword. In my own newsgroups, I know that the anonymity I freely offer people who post helps them to say what they want to say. And I want to know what they really want to say. I don't want to hold them accountable. I don't want anyone to be able to hold them accountable. So it's completely cool with me if unknown people are posting things because it's a good thing. At the same time, in the past, as I've mentioned, we've had trouble being trolled by people who were, in everyone's opinion there, abusing that anonymity and causing problems as a consequence of

posting content they wouldn't have posted if their identity was enforceably known, which it isn't. So it is a mixed blessing, but it is a powerful technology that exists, and I wanted to talk about it.

Leo: And I'm glad you did. And it's a tough one. I just don't know what the answer is, yeah.

Steve: I know. Okay, so TOR. Let's talk about TOR because...

Leo: So that was Freenet, all right.

Steve: That was Freenet, Freenet being a distributed, anonymizing, encrypting database that runs on Freenet nodes scattered far and wide, designed to allow people to anonymously, securely share any content of any kind whatsoever. And they've achieved it.

Okay, TOR is different. And TOR is something that interests a lot of people. We talked about the vulnerability of using a single proxy as an anonymizer because in fact traffic analysis, while complicated, like in cryptographic terms it's trivial. If you take the case of one user using a proxy, it's obvious who they are and what sites they're visiting because anything they do is being done on their behalf by the proxy. Okay, so now that's easy. Now two users are using the proxy. Well, it's a little more difficult. But by looking at the timing of the arrival and departure of packets and the relative sizes of the packets, you could still probably disambiguate the actions of two users across a single proxy. So now we increase it to four and five and six and so forth.

Well, okay. Clearly the more active it is, the more difficult it is to disambiguate queries. But it's not difficult on, like, a really, really, really hard scale. Also you have the ability of storing the traffic and doing an analysis of it after the fact. So you could just capture a huge blob of traffic and then take your time analyzing it to any level of detail needed in order to make your determinations about who was making queries where. The point is that using a single proxy is just not sufficiently anonymous. There isn't a way to make it work well enough.

So knowing that, this notion of a network of proxies was created. And that's what TOR is. TOR stands for The Onion Router. And as I got into this, I realized, okay, I really have to explain how this works because it is so cool. I mean, it is very clever. At the moment there are nearly a thousand onion routers operating in a network. The EFF, the Electronic Frontier Foundation, has been a major sponsor and funder of this because they want to promote this notion of anonymity and actually promote the actual execution, the availability of anonymity on the Internet because they believe in free speech and think that this is something that is worth making sure we don't lose.

So we start with a massive network of onion router nodes. Somebody wants to create a connection through this network to a remote server. They want to use it for web surfing, they want to use it for sending messages, for email or whatever. The TOR system, which is in its second generation now, is a general purpose TCP conduit. So instead of being protocol-specific, for example like a proxy might be an HTTP web proxy and proxying web requests, the onion router system is a general purpose TCP transmission system, so you're able to potentially run any protocols through it that you want to. The originator of a connection chooses at random some number of onion routers that are in the network. And these things, since there's nearly a thousand of them, they're scattered far and wide all over the globe. Now, the more you choose, the more hops your traffic will have to go through, so the greater latency you'll have in your communication. So...

Leo: That's a complaint people always have with Anonymizer, too, which works in a similar fashion, that it slows you down.

Steve: Right, well, and there's just no way around the fact that it's going to slow you down. One of the really important features of this system is that, by going from hop to hop, where these hops between routers have been carefully designed – and that's what I'm going to talk about next is exactly how this works. By going from hop to hop with this protocol, this onion routing protocol, that's where you get the power of real anonymity. And the system is robust, even in the face of many routers being compromised. I mean, bad people, governments, anybody who was not acting in the best interests of the people using the system, could set up a malicious onion router. This system, however, still functions robustly, even in the presence of malicious onion routers that are deliberately being run in order to try to break the system.

Leo: I suppose there's no way that they could prevent that. I mean, people are going to set up malicious onion routers.

Steve: Well, yes. And in fact...

Leo: Or spying onion routers.

Steve: That's one of the attractive things about this is that, because you've got nearly a thousand of these, and they're in all kinds of different countries, they're being run by, you know, many of them by really trustworthy organizations where, you just, like, okay, I trust the EFF to be – really to be on my side. But, for example, a company like Anonymizer – and I don't mean anything against them. It's a great service they provide. But if the government issued them a subpoena, and they're the sole controller of their anonymizing service, they could be forced legally to do something they would rather not do. The beauty of this massive network of nodes controlled by all kinds of distributed entities all over the globe and foreign countries, and the need not to trust individual nodes, it really is the perfect solution. So the way this works is, somebody who wants to create a circuit through multiple onion routers chooses the routers they want to use and the order in which their traffic is going to move from router to router.

Leo: There's probably a default setting, right? I mean, you don't have to go through this every time.

Steve: I don't mean that the user manually chooses. I mean that the system chooses for you. But I'm sort of operating down at the protocol level. The way the system works is, a random set of routers is chosen. All routers have a public key pair, that is, they have their own private key that only they know, and they have a public key. And as we remember, the way this functions is something can be encrypted using their publicly available key which everyone can know. And once encrypted, that data can only be decrypted using the matching private key that that router never divulges. That's its secret key that matches its public key. So all routers have these asymmetric or public key pairs. What happens is, the user starts at the far router. The user is going to build what's called an "onion." And the term "onion" is used because we're used to onions being created in layers.

Leo: Having skins, yeah.

Steve: Exactly, having skins, and you successively peel off the layers of the onion.

Leo: An ogre is like an onion.

Steve: So conceptually...

Leo: You didn't see "Shrek," so you wouldn't...

Steve: I did not see "Shrek." He's a big green thing is all I know.

Leo: But he says ogres are like onions, they've got many layers.

Steve: Okay. So...

Leo: And so are TOR routers.

Steve: So are TOR routers, yes. So the user preparing the onion takes the...

Leo: I really threw you with that, didn't I. I'm sorry. I won't say anything more about ogres and onions.

Steve: This is really complicated, so I want people to pay attention. It's really complicated, but really good. So the user takes the public key of the last router in the chain that they've chosen. And they take a randomly generated symmetric key, which is going to be used to decrypt the traffic it receives. And they encrypt that symmetric key with that last router's public key. So now they have a blob that contains an encrypted symmetric key that only the last router is able to decrypt using its private key.

Leo: That makes sense, okay.

Steve: Now they take that blob, they come up with another – they just randomly generate another symmetric key which the second to the last router is going to use to decrypt the traffic it receives and the symmetric key they gave – I'm sorry, yeah – to decrypt the traffic it receives and the identity of the next router that is the last router in the chain, which is the one to which the second to the last router will send its traffic. So there's another symmetric key, and the routing information for the second to the last router, and the blob, which we have from our first encryption. We encrypt this entire thing using that second to the last router's public key. Now our onion has two layers.

Now we step back to the third router. We randomly create another symmetric key which it will use to decrypt the traffic it receives, and we give it the routing instructions for the router second to last, meaning the one it will forward its traffic to, and that entire blob we had before, that two-layer onion, and we encrypt the whole thing using that third to the last router's public key. Now our onion has three layers. And we continually step backwards, building this onion, this nested multilayer encryption, each layer encrypted with a successive router's public key,

which only that router knows how to decrypt, and each layer containing a symmetric key which was generated randomly by us at our end. When we're finally done, we have this – we've stepped back through the whole chain, back to the onion router that we will be using to inject our traffic.

And so now we've got this thing called an "onion." We give it to that first router. It uses its private key to decrypt the outer layer of the onion. Only it is able to do so. Nobody listening in on the traffic is able to see that happen. Only it internally knows how to use its private key to take the wrapper off of the onion. When it does so, it has an opaque blob that it cannot decrypt because only the next router in the chain has the matching private key to decrypt the next layer. But when it decrypts it, it finds a symmetric key which it will use for decrypting traffic it receives and the routing instructions for the next router. So even somebody – there's no way to even know by looking at the onion what the path will be. Only the router that decrypts its layer knows the identity of the next router in the chain. It knows nothing about any other routers in the chain. It doesn't even know how many other routers there are. It knows nothing except its own local, I'm going to receive data, decrypt it with a...

Leo: Very clever.

Steve: ...and where do I send the result.

Leo: That's very slick. That's so slick.

Steve: Oh, it's so cool. So of course this thing then, it now knows where to send the remaining blob. It knows which router to send the remaining blob to, the so-called onion. It sends it to that router. That router receives it, uses its private key, which is the only key which can be used to take the next layer off the onion. From that it gets its symmetric key for this connection and the routing instructions for the next router, and it forwards this thing down. So this onion, which was built once in the beginning, it then moves through the network, basically informing each node of the onion router system, only the information it has to have, I mean literally only the information it has to have, how to decrypt what you receive, who to send it to. And each onion router only knows one step in the chain and no more.

Leo: That seems impervious to any kind of – including a corrupt router.

Steve: It is just deliciously robust, Leo. I just love it.

Leo: However, it seems like there's also a lot of overhead. You've got a lot of encrypting and decrypting going on.

Steve: Well, now, that's the other beauty of this, is remember that asymmetric encryption is very slow. Public key crypto is on the order of a hundred times slower than symmetric. So all of that work only needs to be done once. What we just did was we established a circuit. We have not sent any data yet.

Leo: Oh, I get it, okay.

Steve: The onion is not a data carrier. It's just a circuit creator.

Leo: It's a route, basically.

Steve: Exactly. It is an incredibly cool, secure route that also manages to supply symmetric keys to every router along the way.

Now we want to send data. To send data, we take the actual data we want to send. The first thing we do is pad it out to a fixed size. That's one of the other cool things these guys did. They recognized that traffic analysis could use the packet size for disambiguation, as we saw in the simple single-proxy case. So they've completely solved that, so that no matter how big the packets actually are, we're going to pad them out to full size so that all packets moving among the routers within the onion router system are the same size. You can no longer use size to give you any clues because they're all the same.

Leo: PGP does that, too. That's a great technique.

Steve: Yes. So the next thing we do then is we take the data we want to send, we use symmetric encryption – remember, that's very fast. Symmetric encryption is very, I mean, it's like not a concern anymore, it's so fast. We first encrypt it with the last symmetric key, that is, with actually the first symmetric key we generated, which we gave to the last onion router in the chain. Then we encrypt that with the second to the last symmetric key. We encrypt that with the third to the last symmetric key, and so on, working our self back up the chain. So we have something still the same size, that is, because symmetric encryption doesn't change the size. Remember, it just flips the bits. So we've got something which has been encrypted N number of times where N is the number of routers in our onion routing chain.

We then give this blob to the first onion router. The last encryption we did was with its symmetric key, so it's able to decrypt it. But remember, all it's doing is decrypting a layer of encryption. It's still a blob, that is, it's still encrypted N minus one times, using keys it has no knowledge of because those keys were buried in layers of the onion which were encrypted using the public keys of the other routers that can only be decrypted using their private keys that no one has any way of knowing. So it takes this and simply – so it runs a very fast symmetric decryption on what it receives and forwards it to the next router in the circuit because it's maintained a circuit awareness of what its next router is. Remember, it knows nothing other than the next router. That router receives it, sees what circuit ID it is, looks it up in its table. It uses its symmetric key to again perform the same decryption on this data packet, which it then forwards to the next router.

The beauty of this is there is no way for anyone looking at an onion router to have any clue what's going on. Not only are the packets all the same size, but because a symmetric decryption occurs between incoming and outgoing data, and we know that state-of-the-art encryption just looks like randomness, there is no correlation between the data of the packets coming in and going out. It's just, you know, you get – it looks like...

Leo: It's just noise.

Steve: ...packets of random, yes, packets of noise came in; packets of noise went out. And because this is a richly interconnected system, packets are coming in from all these onion routers all over and going out to all these onion routers all over. But there's nothing to help you determine which one is which, I mean, like where the circuits are. There's just no way to figure that out. So when this data packet finally gets all the way through its multiple hops, it is finally decrypted with the last symmetric key by the last node, and that is again the text or the data that you originally put in at the front end. And baby, that is anonymous.

Leo: Wait a minute. That's the title of this show: "Baby, That's Anonymous." And by the way, you've more than made up for – if anybody thought last show was fluffy, you've more than made up for it with this one. But once again it's very elegant to see this really nifty solution to a complex problem.

Steve: Oh. I just, I mean, it's one of these things where – and I promised our listeners this a year ago, that once we had the foundations of symmetric and asymmetric crypto under our belts so we could talk about the things you could do with it, this is the kind of stuff you can do. I mean, it is so neat. And there's thousands, well, not thousands, there are – it's more than many hundred. It's, I think, nearly a thousand of these onion router nodes. The system is called TOR, The Onion Router. It is slow. I mean, it's not like going to Microsoft.com or Yahoo! or Google, where you're making a point-to-point connection. But they know everything they care to know about who you are. The tradeoff is you need to go through multiple hops. We've seen why multiple hops are necessary, why just running through a single proxy server doesn't really give you robust anonymity. But this onion routing system does. And it is just elegant.

Leo: Yes. Tor.eff.org is a good place to go to find out more and actually to start using the TOR system.

Well, now I feel – the thing about TOR, unlike Freenet, you're not really volunteering, unless you volunteer to be a TOR system, you're not volunteering for anything you might not be comfortable with.

Steve: Well, yes, you're right.

Leo: And you can use TOR completely freely and safely.

Steve: Yes. You can certainly be an onion router, in which case you would be offering your node to enrich the network. You'd be offering your bandwidth also.

Leo: But you don't have to to use it.

Steve: That is correct. You are able just to be a user of the network. You would also – I was going to say, you would be offering your bandwidth. You would be offering your node to enhance the anonymity-creating ability of the network. But you would not be offering any storage space.

Leo: You don't store anything.

Steve: Exactly.

Leo: Hey, I want to thank you, Steve, for clarifying this and for talking about it because I think it's very important. And I'm a big believer in freedom of speech and privacy, and I think these two technologies are very important to preserving that.

Steve: I guess my position is, if somebody needs it or wants it...

Leo: There it is.

Steve: ...it's cool that it is available to them, yes.

Leo: That's exactly right. What else is available? GRC.com, a great place to go to protect your system, whether it's with ShieldsUP, the free tester that'll test your router or your firewall, or many of his free programs – Shoot The Messenger, DCOMbobulator, UnPlug n' Pray, and on and on and on, and a new one coming up any day now.

Steve: I get to work on it now.

Leo: GRC. You've got a couple weeks to work on it, yeah. GRC.com. That's where you'll also find the show notes with the 16KB versions for the bandwidth impaired and the transcriptions by Elaine so you can read along. This is one you probably will want to read along to, I have a feeling. And SpinRite's there, too. That's Steve's daily bread and butter. He has been writing that, wrote that program about 15 years ago now? You're at version 6? And it's still going strong. Still the best, I think there is no question, the best disk recovery and maintenance utility on the market today. If you have a hard drive, you need SpinRite. Any good letters lately from SpinRite?

Steve: I got one cool one. It's from a guy who said, "I'd like to thank you for SpinRite 6. I was handed a computer with a completely dead hard drive from a Catholic parish." He says, "I tried two other tools which ran in Windows. Both locked up at about 60 percent into the test." He said, "I then put these aside and bought SpinRite. I had been meaning to buy it for myself anyway, since I'm the one people seem to come begging to around here.

Leo: And that's who should have it, the people – you know, that's why I have it.

Steve: Yeah, yeah. He said, "It took SpinRite a long time to complete. But when it was done, I was able to get the data off the drive. There were some corrupted files, but the parish said I had the most valuable ones." He said, "This may sound strange, but I was also able to prove to them that, even though we were able to get the data back, the drive should be replaced. Watching SpinRite work on the drive also helped me to convince them to take the time to back up."

Leo: Well, there you go. Well, that was worth it, then.

Steve: So it was a win for everybody.

Leo: Well, Steve, I thank you so much. We will adjourn and then restore, return to our regular security meeting in a week, on Thursdays, as always, by hook or by...