SECURITY NOW!

Transcript of Episode #274

Benchmarking DNS

Description: After catching up with the week's security updates and news, Steve formally unveils GRC's latest freeware, the DNS Benchmark. Steve explains the value of the program's many features and discusses the operation of this "long time in coming" freeware offering.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-274.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-274-lq.mp3

Leo Laporte: This is Security Now! with Steve Gibson, Episode 274, recorded November 10th, 2010: Benchmarking DNS.

It's time for Security Now!, the show that covers your security and privacy online. And the man who does it for us, the great Steve Gibson from GRC.com. Hey, Steve, how are you today?

Steve Gibson: Hey, Leo. Great to be with you again, as always. And you promised me that the recorders are running for this; right?

Leo: Everything is recording. Well, let's put it this way. If you are hearing or watching this show after the fact, we recorded it.

Steve: I guess by default, yes, sir.

Leo: Actually we missed - last week I [indiscernible], I was kind of flustered, and I forgot to record the show. We did a whole show. But one good thing, because we do stream everything live, one of our partners, Justin.tv, records everything. Actually two of our partners do. BitGravity also records everything. So we're able to go to their site and get, not a super hot, not the highest quality, but as good as you would see if you were streaming it quality. And so we were able to put out the show...

Steve: Yay.

Leo: ...despite Leo's stupidity. We had, you know, we had a backup system, and I don't know, I guess - that's embarrassing, isn't it. The system that we had, which was recording everything, has failed, and we didn't put another one in place.

Steve: And I think I heard you mention yesterday that you're not now mentioning that by name because the people presumably who created it are a little sensitive to it being dissed for the fact that it keeps [indiscernible].

Leo: Ah, yes. We're talking - that's the "thing that we use."

Steve: The thing that shall not be mentioned.

Leo: The thing that shall not be named. I just, you know, I realize...

Steve: Is that the Tri-something?

Leo: Yeah. We use this to switch. That's how I switch. And it's really remarkable, I mean, given the cost, what it will do. And we couldn't really do these shows without it. However, because it's on 24/7, it does crash from time to time. And I think the people who make it are a little sensitive to the fact that it crashes in public because it's not really designed to be used the way we use it, full-time, 24/7. It's designed to switch a two-hour show, then go to bed for the night. And we really use it like a hardcore studio.

And actually, what people don't know because TV seems so reliable is that television studios go on the fritz all the time. That's why TechTV had dual studios. We built two one-million, I think it was $1.5 million, studios for redundancy. So if one goes down, you stay on the air. If you're a network, you have to stay on the air. We're running the whole thing on a $10,000 TriCaster. But we are - eventually we'll get some redundancy.

So I'm excited about this week because finally we're going to talk about something you've been working on for some time.

Steve: Well, it's been a - it's a project that actually had its beginnings a long time ago. Back in 2001, shortly after the 9/11 attacks, I was asked by the White House to explore the idea of a communication system for the Internet, which didn't exist at the time. Well, the Internet existed, but something that could sort of, like, put out a flash announcement, an emergency sort of thing that could somehow deliver a message across a huge number of, or to a huge number of devices in a very, very short time.

Leo: That was kind of prescient. I mean...

Steve: Much faster than email, yeah.

Leo: We could use that right now. Everybody now carries a portable device with them.

Steve: And I remember you and Mark Thompson and I were walking around the streets of Iowa on October - I mean, after the September attacks, and in fact for that first Gnomedex that I keynoted. And we weren't even sure if Gnomedex was going to happen because the airlines had been shut down for that purpose. But and so what happened was I created something called DNSRU, the DNS Research Utility. And I worked with a large group of people in GRC's newsgroups to experiment with the way DNS was working. And although it wasn't directly relevant to the research I was doing, I sort of, while I was there working with DNS, I wrote sort of some benchmark-y stuff for DNS which did a sort of an initial job of benchmarking DNS servers. And when I had finished with that work, I just sort of left this DNSRU in the state that it was, which was sort of, like, it wasn't ever meant to be polished and made public or anything, it was just sort of an internal tool.

But the people at GRC, hanging out in the newsgroups, just they kept it alive. And, like, years would go by, and every so often someone would say, "Gee, you know, are you ever going to finish that?" And I said, "What? Why?" Well, and the answer was always, "Well, there's nothing else like it." And so, like, the people who, like - we had switched over to using Level 3's famous DNS servers - 4.2.2.1, 4.2.2.2 and so forth, up to .6. And people who were sort of more on the guru side, they had continued to use this thing that was never meant for primetime because that's all there was. There was no other way to do this.

And so a couple years ago, when the whole Dan Kaminsky DNS spoofability thing happened, and I jumped onto creating a facility for looking at how spoofable name servers were, I thought, well, I guess I ought to finish that old DNSRU thing. Well, what happened was it acquired a life of its own, and it became a really beautiful piece of work, a piece of freeware that I'm really, really proud of. And so I wanted to talk about the task of and the technology of benchmarking the domain name system and sort of introduce this to our listeners, which is now available for everyone for free.

Leo: That is exciting. Its debut in just a moment.

Steve: Yeah.

Leo: Well, I'm looking at your show notes, and we have a few updates to talk about.

Steve: Yeah, not too much. We are at or just past the second Tuesday of the month, which of course we all know means that Microsoft will have done something.

Leo: Not much this time.

Steve: Not much this time. And in fact the one thing that they didn't do is the one thing that they really wish they had, which was the very bad zero-day vulnerability in IE is becoming increasingly widespread. And we got no fix for it. I'm not surprised, though, because this caught them by surprise just, like, almost on the eve of this second Tuesday of November. So there's a bunch of fixes for Office. One is critical, where just previewing email in the preview pane of Outlook could allow an infection to load code and commandeer someone's machine. So that's fixed.

Unfortunately, the big zero-day flaw, as I mentioned, in IE is still not fixed. Several people, including Brian Krebs, noted that this has now been moved into one of the more popular all-in-one hacker kits. So it's now in a toolkit. So the use of this unpatched zero-day IE flaw is expected to increase greatly above the few targeted attacks where it was seen before a week ago. And of course that's what we expected. IE6 and 7 are vulnerable. IE8 is not so much vulnerable. Technically the flaw exists in IE8. It involves a problem in the parsing of CSS, of the cascading style sheets on websites. There's some token parsing problem.

And Microsoft has one of their Fixit updates, actually they have two different Fixit buttons for this. But really, if you upgraded to IE8 - which I would at this point. It's mature enough. We're in IE9 beta right now. I would say anyone using IE6 or 7 should just update to IE8. It runs data execution protection, DEP, by default. And that's enough to thwart this problem. Even though technically the browser still has it, it isn't exploitable due to DEP locking this down. It's doing exactly what it was intended to do.

So if for some reason you can't go to IE8, then you can turn on data execution protection for IE6 or 7 by manually turning it on. Or you can use Microsoft's little Fixit thing to do that. They also have kind of a funky other Fixit that involves CSS somehow. But the problem is it may break some websites that you go to because they've, like, turned off the parsing for some aspects of CSS. So I'm not so jazzed about that. Of course you could also just not use IE, which would be a fantastic solution. Switch over to Firefox or Chrome. I would say that's an even better solution. But for the sake of completeness, Microsoft's Knowledge Base that has these Fixit tools, it's support.microsoft.com/kb/2458511. So again, that's support.microsoft.com/kb/2458511. So that's where the little Fixit things are, if you're stuck with using IE6 and 7 for some reason.

And meanwhile, Chrome continues to sort of silently slide itself along. Last week remember that I had noted that it had moved itself up to a big long number ending in .41. This week we're at .44. So the SANS Security Institute newsletter reported: "Google has released patches for multiple unspecified vulnerabilities in its browser. The vulnerabilities include two use-after-free errors, two unspecified memory corruption errors, a bad cast, an invalid memory read, integer overflows, and an out-of-bounds array access. These vulnerabilities exist in [the] libvpx [library] and the code for handling text areas, XPath, fonts, and [Scalable Vector Graphics]. Some of these vulnerabilities may be exploitable for code execution if an attacker can entice a target to navigate to a malicious site."

So it's like, okay, well, so this is sort of the Chrome model is they're just quietly fixing these things in the background. And I note that whenever I fire Chrome up, it knows what the latest is. I mean, it's already updated itself to the latest and greatest. So this is a different approach than Microsoft's lumping things together and dropping them sort of en masse on a monthly basis. Google is just keeping Chrome up to date. Interesting, Google paid a total of $8,674 U.S. for 11 of those 12 vulnerabilities.

Leo: Really.

Steve: Broken out into the researchers who reported them. Google has, as I'll mention in a minute, a bounty program where they pay developers for responsibly, which is to say quietly, reporting bugs to them that are found. And they give them public credit and dollars for doing so. Which is, I think, an interesting model that seems to be working for them.

And speaking of Google, there is a proof-of-concept demonstration out for Android's browser, the down version one, Android v2.1 and earlier. And although we are now at 2.2, it turns out that at least two thirds of all instances of Android are still at 2.1 or prior. And so a security researcher at a company called Alert Logic essentially got annoyed that this has been a long-known flaw in WebKit, which is being used by the Android browser, which has been fixed in 2.1, but it isn't fixed - I mean, sorry, has been fixed in 2.2 of Android, but not in down version instances, of which, as I mentioned, more than two thirds of the market still is.

And so this guy is saying, look, I want to put some pressure on Google to - well, and through the whole chain - to get these old problems fixed because a substantial market are still using those. And in fact Gartner just came out with a report showing that Android is now the No. 2 mobile OS. Symbian still has the first position. But Android - I think they had, like, 38 percent of the market. But Android is now up to 25.4, I think it was, with iOS in third place. So it's really doing well.

And also, a company called Coverity apparently took a developer's build from the HTC website and ran their automated kernel-testing tool against it, and exposed hundreds of flaws, 88 of which they said were high risk. And these are, like, buffer overrun, uninitialized variables, just sort of runs the gamut of different problems. They've informed Google of all these problems and given Google 60 days to respond, saying that they will release nothing additionally about this for the next two months, giving Google a chance to look at these and figure out what they want to do. But again, they're saying that they found a whole bunch of flaws in - and it's interesting, it's in the portion of Android which was inherited from Linux more than the newly written code, which seems to be in much better shape.

And, finally, Google has expanded the Bug Bounty program. I was just mentioning it in Chrome. They've expanded it to cover Gmail, YouTube and Blogger, that is, those web-based facilities; not Android, Picasa, and Google Desktop. But anyone finding a flaw, and reporting it responsibly, in Gmail, YouTube, and Blogger, and that means cross-site scripting or any sort of CSS vulnerabilities, anything of any sort that is found, Google will pay $500 for, in addition to giving public recognition. And in cases where the flaw is really severe or particularly clever, they'll pay up to $3,133.7, which seems like a strange number except that that's "leet."

Leo: 1337.

Steve: So they said, okay, we'll make it worth the people's time, if they come up with something really good.

Many people tweeted me today something that hit the news, which I actually knew about last week. The folks at Anonymizer are in the process of finalizing something called "Nevercookie," which is an add-on to Firefox to deal with the Evercookie exploit, which our friend Samy Kamkar we've talked about now several times. Of course the Evercookie was a disturbingly capable identity-tracking technology that used JavaScripting in all kinds of ways to squirrel away some sort of identity about a user everywhere possible in a browser. And so the Anonymizer folks said, well, okay, let's block that with the Nevercookie.

I have a copy of it. And my preliminary analysis says, yes, it's doing the job, although I told them I wanted to do a detailed packet analysis to get a better look at it. And they expect to be releasing it here within the next few days. Maybe we'll be able to announce its availability next week. So it's just an add-on that will be available; as soon as it's available you will be able to add it to Firefox. And it just forecloses all of those various things that the Evercookie is doing. Which sort of seems like a good idea.

Leo: No kidding, yeah.

Steve: Firesheep continues to be in the news. I did a little refresh of the download page, and every time I look it's gathered another thousand downloads. I think we're now at about 703,000 downloads.

Leo: Geez.

Steve: 703,000. So we're well past the half million point, approaching three quarters of a million, and we're going to get there at the current rate. Probably by the time our listeners hear this I would imagine we will be at, well, maybe almost. But 703,000 at this point.

Briefly, last week, someone had produced a piece of response software which attacked Firesheep in someone's machine. That is, it somehow put out packets on the network which, like, crashed it or something. But by the time - and I read about it a couple places. By the time I went looking for it to figure out what was going on and what it was about, it had already been taken down. It had already disappeared. So I think whatever, I mean, it sounded like a bad idea. And I think it was so bad that it didn't last long.

However, there is something which is a clever response to Firesheep known as Blacksheep. And Blacksheep is also an add-on to Firefox which detects the presence of somebody, anybody, using Firesheep on the network, like on the hotspot where you're located. And the way it works is clever. So it's called Blacksheep. It's a Firefox add-on. When you run it, it periodically, and you are able to configure how often it does this, it periodically creates a fake credential, that is, a fake account which it puts out onto the network, knowing that Firesheep will pick up on it.

Leo: With a bogus cookie.

Steve: Well, exactly. And so, well, and then what Firesheep does is, it's as if somebody had just come onto the network, for example, some new Facebook user. So Firesheep, as Firesheep does whenever it sees that, it attempts to log onto that person's Facebook account to get their picture from their Facebook page to post it in the little buddy list that Firesheep maintains. So what happens is Blacksheep detects that attempt to log onto the bogus account which it deliberately created and put out there sort of as bait. So basically Firesheep falls for the bait. Blacksheep detects that something fell for its bait and alerts you that somebody somewhere on the network is running Firesheep.

So the bad news is, of course, if you were a Facebook user, or Twitter or whatever, there's still a good chance that, in logging onto the network, getting onto the hotspot, you may have already exposed yourself. But at least it does let you know that someone's using Firesheep. And if you aren't a user of those services, but you're just sort of curious about the prevalence of Firesheep, you could easily run Blacksheep for a while, while this is all going on, just to have a little pop-up notice that, hey, by the way, somebody's running Firesheep somewhere on this hotspot. So I thought it was a clever hack.

Leo: Yeah, I like it, I like it.

Steve: And Microsoft is the first, that I'm aware of, responder to Firesheep for Hotmail. They have just announced on their blog yesterday, and it is now available, an option to turn on pervasive SSL encryption for Hotmail. So the Hotmail no longer...

Leo: Yay, yay, yah.

Steve: Oh, yes. Hotmail no longer only uses SSL for logging on. You are able to turn it on so it will persistently use it and protect all of your email traffic, all the time.

Leo: Excellent.

Steve: And I had a nice little note from a Security Now! podcast listener, actually a Jonathan D. Kramer, who's the director of technology at St. Mary's High School in Manhasset, New York. He said, "Steve, I just wanted to thank you for your wonderful product, SpinRite 6. I'm a loyal listener to your Security Now! podcast and have heard you talk many times about SpinRite. Well, this week it happened. After a weekend of thunderstorms and wild weather, I arrived at work to find a colleague at my door, sweating and explaining that his computer was dead. When I went to investigate what was going on, I was horrified to find, or should I say hear, what his computer was doing. There was chirping, buzzing, and downright singing coming from his hard drive."

He said, "Like Superman, I put my hands on my hips and said, 'I know what to do.' Of course the fool had no backups. Well, I downloaded the software, let it run, and it brought back his desktop. SpinRite literally saved years' worth of his work. Thank you for what you do for the industry, and keep up the great work. Jonathan Kramer." And thanks, Jonathan, for the great report.

Leo: All right, Steve Gibson. I'm ready to hear. This is a program you've been working on since 2001.

Steve: Well, certainly not continuously. But, yeah. This is sort of - this is something that would not die. And I was just, because I was back working on DNS stuff, when I was working on the spoofability analysis system that we'll be talking about next, and shortly, and because users, I mean, like GRC's newsgroup people had fallen in love with this funky, not-really-ready-for-primetime DNSRU thing that just sort of did some benchmarking on the side, I thought, okay, well, I'd invested already in a lot of that technology. If I didn't ever finish it, then the world would never have a really good DNS benchmark. And so I decided it is a useful thing to do. There are expert users who want to know how fast Level 3 servers are compared to their ISPs. We know that ISPs often have slow DNS servers because DNS is just sort of something you stick in a closet and you don't think about it, unfortunately, which is in fact why so many DNS servers are still exploitable for, like, the Kaminsky spoofability problem.

Leo: Shamefully, but...

Steve: Shamefully, yes. But also, for example, we now know that there are an emerging set of alternatives. There's OpenDNS, which provides true value-added DNS services. The question is, okay, how fast is that? Am I sacrificing speed if I switch to OpenDNS, compared to my ISP's servers? And we know that the Sunbelt Software people have got a security-enhanced DNS where it blocks you from going to known malicious websites. Symantec is getting into it with something that they're calling Norton DNS. And there's UltraDNS.

And then we have the other problem that we've talked about where many ISPs are seeing DNS as a new revenue-enhancing model, where if you put in a typo, you don't get an error, but you get redirected to their own up-selling marketing page, where they're trying to turn your typos into some sort of revenue model for them. So what they're doing is, they're redirecting errors. So I thought, hey, it would be nice to be able to detect that, as well.

So it made sense to me that, I mean, I could understand why GRC's newsgroup mavens were saying, hey, you know, we'd really like a finished DNS benchmark product. There seemed to be a lot of different things to do. And then we were talking recently about this whole DNS rebinding problem, the idea that a script in your browser could fool your browser's protection which restricts what that script can do by causing a remote DNS server to return your own IP as if it was part of that script's domain. And in the process, that would allow the script to have access to your local resources, maybe your router or your own machine, which would normally be reserved only for it to act with its own source domain. So this rebinding protection, I realized, hey, that's something that I could detect from the benchmark side. So I ended up building all of this into this nice piece of freeware.

So it runs natively under Windows. But we have a whole bunch of people who are UNIX users, who were involved in working with me on the development of this, over on sort of like on the testing and kibitzing side. And so I spent some non-insubstantial amount of time making sure that this would run under Wine, and run under Wine on the Mac. So you can use it if you're a Mac user, and you have the Wine-is-not-an-emulator emulator for Windows. And it runs under Wine under Linux with no trouble at all.

I did write the whole thing in pure assembly language. It is, with all the features that it offers, 163K in size, so really small for - when you look at what this thing does and see it running, the idea that it's 163K, I mean, JPEG images are much bigger than that these days, it sort of reminds everyone how much can be done in assembly language and how inefficient so much "modern" software has become. It's also fully scriptable, so that people could, if they wanted to, like, run it in the middle of the night, run it a couple times a day. It's able to export its results to a CSV, to a Comma Separated Values file, so that you can use it to, like, log the performance over time. It's just got features coming out of its ears.

But primarily what I wanted to do was give users a useful sense for how fast DNS is. And there are several different parameters for DNS performance. There's how quickly you get a response when the thing you're looking up, the domain name you're looking up, is already in the DNS server's cache. We've talked about DNS caching a lot, the fact that in fact this is the whole problem with spoofing is, if a DNS server can somehow have its cache poisoned, then it's got the wrong IP for a domain that you're looking up. And so when you go to, for example, Amazon.com, your browser could actually be looking for www.amazon.com, receive the wrong IP, and be taken to somewhere completely wrong. And so that's one of the main ways that very strong phishing can operate is this kind of cache poisoning.

So we know that resolvers cache. The reason they do that is that they tend to be close to you, network-wise. That is, most ISPs, probably all ISPs, provide a DNS server, the idea being that, since it's your ISP, in terms of networking distance it's very close to you. And since the ISP has this hopefully large server, all of its users are using the same server. So you would imagine that Amazon.com is already in the DNS server's cache because somebody else who's also a user among many users of a given DNS resolver, they will have within, like, for example, the last day asked for Amazon.com. If it wasn't cached then, Amazon.com would be cached, the IP for Amazon.com would be cached, so that your request would then come from the cache.

So one of the things that this Benchmark does is it makes two queries for, for example, Amazon.com. It makes the first one in order to make sure that it's loaded into the cache, and ignores the length of time that query requires. Then it makes a second one to essentially know that that query will be in the cache. It turns out that there is a bit in the response which indicates whether it was from cache or was not. So the Benchmark verifies that that second query did come out of the cache because technically you could - that first query could occur just as the copy was expiring from the cache, meaning that the Benchmark would be fooled by the second query being cached rather than non-cached. So I make sure that that doesn't happen.

And so that gives us a measure of how fast the performance is out of cache. And we really weight that much more highly than non-cached queries because the theory is you're using a large DNS server, as I said, that many other people are going to be using. So primarily I would imagine the huge majority of queries that an actual user is making will be coming from the DNS server's cache. But it's certainly the case that some wouldn't. That is, if you make a request for some obscure domain that is either - no one else has ever asked for, or they're asking for so infrequently that it tends to expire, or it may be that the domain for whatever reason has a very short expiration. For example, there are domains that expire every hour because they tend to be victims of denial of service attacks, and so they're having to move their IP around from time to time. Therefore they don't want DNS servers to keep an old copy of the domain records current for a long period of time.

In any event, what the Benchmark does is it deliberately asks a dotcom server for a known invalid machine. For example, it might ask of Amazon.com, it would look for the domain, just some gibberish, 13 or 14 characters of gibberish, .amazon.com. It asks a DNS server for that, knowing that it cannot be in the cache because it just made up this machine name. So your DNS server looks to see whether it's there. It isn't because it's a completely unique name, which forces the DNS server to go ask the Amazon.com server whether it knows the IP for this. The Amazon.com server will say, no, that's an invalid name. But that allows us nevertheless to measure the length of time required for this particular DNS server to reach out onto the Internet for something not in its cache to any of these various dotcom servers. So that's the non-cached lookup parameter, which the Benchmark measures separately.

And finally, I also ask for - I have the Benchmark ask for a gibberish name dotcom, which is to say a primary dotcom domain that does not exist. What that forces it to do is to go ask the dotcom servers, which are one level up in the DNS hierarchy, whether they've ever heard of this dotcom domain name. Which again, they won't have, because we just made it up. But again, it gives you another measure.

Essentially, both of those last two, the non-cached lookups and the dotcom lookups, are a measure of how well connected this DNS server is to the Internet because on one hand you want to know how well connected it is to you. And the cached lookup measurement gives you that. But you also want to know how well connected it is to the Internet. It could be very close to you, for example, being your ISP's DNS server. But it might have a really clogged, buggy, slow, whatever, bad connection to the Internet, meaning that anytime you ask it for something it doesn't know, it's going to take a long time to get that reply back.

So the Benchmark measures essentially three parameters of performance - the cached lookup, the non-cached lookup, and the dotcom lookups, independently - and shows you in a very nice graphical display how this group of name servers that it checks compare with each other. It also measures the reliability of all of these name servers in response to the queries. Basically, it knows how many queries it's issued to each name server, and it looks to see how many responses it gets, looking to see whether the name server may be dropping them, and having a problem with reliability.

So we have a built-in list of, I think it's about 70 sort of well-known name servers, which we test against 50 very well-known domains. So, for example, I actually got the list from Alexa's list of top domains: Google.com, Yahoo.com, YouTube.com, Live.com, Facebook.com, MSN, Wikipedia, Blogger, MySpace, Yahoo.co.jp, Baidu, Google.co.in, Google.de, Microsoft.com and so forth. So basically the top 50 most popular domains on the entire Internet. There were some that I had to remove because they were sort of questionable domains, things that you might feel self-conscious having a program running in your computer going and asking the IP for. Adult content websites were removed just for the sake of propriety.

So once we've got that, though, there's another question that comes into play, which is I can rank and do all of these resolvers, which are built into the Benchmark. The Benchmark has, as I said, about 50 very well-known resolvers - a bunch that belong to Cox, Google's two resolvers 8.8.8.8 and 8.8.4.4, the OpenDNS resolvers, UltraDNS, the Symantec Norton resolvers, sort of all of the ones that are very popular, all six Level 3 resolvers. The idea being that you want to see how those compare, not only to each other, though, but also to whatever ones you're using.

So the Benchmark determines which resolvers your system is using, and benchmarks those right alongside this other list of very well-known, sort of publicly known popular resolvers. And it ranks the ones your system is currently using relative to all these alternatives to see whether switching to one of these alternatives or one or more might make sense. But the question is, even if we get sort of average values which appear to indicate that one resolver is better than the other, the question is, is it reliably better? That is, is it statistically significantly better?

So the Benchmark goes beyond just taking average values. What it wants to do is it wants to verify that we're sure that the 50 samples we took create statistical significance. For example, say that we took five measurements, and that a given resolver came back with a rating of 50, just 50, where faster is a lower number; and this was, like, 50 milliseconds to respond. If we did five measurements, and they were all 50, well, we'd have a strong confidence that if we took another five measurements, they would also all be 50. So we could say, okay, this is the performance of this resolver with very low sampling error, that is, high confidence that, if we sample again, we're going to get the same thing.

But say that we took five readings that were 10, 30, 50, 70, and 90. Well, the average of those five is still going to be 50. But our confidence is much lower now that, if we took five more readings, like a different five readings, that we'd still get an average of 50. We might get, since we saw that one sample came back at 10, well, we might get 10 five times, or we might get 90 five times because we've seen a big spread in the return. Well, statistically there's a measure known as a standard deviation, which is a measure of the spread of samples around the average. The Benchmark takes that into account, and it tells you whether it would make sense with 95 percent confidence, that is, it's able to say we are 95 percent confident of the following conclusions.

And in fact, one of the nicest things that the Benchmark does is there's a lot of data being presented in bar graphs, and there's also a tab that gives you a sort of a detailed statistical analysis, all the numbers that back up all the data that's being shown. But I wanted to also simplify this for the typical user. So there's a conclusions tab where, once the Benchmark is finished, in English, I basically do all the work for the user of figuring out what this means.

I heuristically, using a bunch of algorithms, write in English a set of conclusions about what the Benchmark found, telling you, for example, whether you're using resolvers which are redirecting errors and not returning them to you, but sending you to a different page; noting, for example, that X number of publicly available resolvers are with 95 percent confidence faster than the ones you're currently using, whether the ones you're currently using are faster than all the known alternatives. Basically, I do all the work of interpreting these results in English so that anyone can just look at the conclusions tab and read down and see what it was that the Benchmark found.

And then lastly, near the point that this thing was finished, Google announced that they were getting into the DNS business with their two resolvers, 8.8.8.8 and 8.8.4.4. They also hosted somebody's program called Name Bench, which was a benchmark that was hosted on the Google code codebase. People thought that it came from Google because it was located there. But it's not an official project of Google. I had the people in the newsgroup who were testing my benchmark program try Name Bench. And the conclusion was that it really was really not ready for primetime. Basically, it crashed everybody's routers, which...

Leo: Oh, that's not nice.

Steve: Not a good thing to do. What happened was it sent out so many queries that it overflowed the NAT routing table in the routers. And it just took - it knocked everybody off the Internet. One thing that it had, though, was pretty cool. Thanks to it having access, I guess, to some resources from Google, it had thousands and thousands of DNS servers. That is, it had a list of some 4,400, I think it was, DNS resolvers. And so I thought, well, okay. What if there's some good ones in there?

Leo: Right.

Steve: So I made an experimental version of the Benchmark, mine, because I normally had it set up so that you could benchmark 200 at a time, figuring, wow, that's more than anyone's going to need. We're benchmarking, I think, 70. But you do have the ability to set up a custom list, like of your own resolvers, that you'd like to benchmark. You may have, like, some corporate resolvers that you want to test. So you're able to add those and then save an .ini file, an initialization file, so that the Benchmark will automatically load those back in every time you use it.

So I made a temporary, just sort of a real hack to expand the number of resolvers that could be benchmarked to 5,000. And this is not something you would ever want to do because it takes a long time to run. But what we found was there were obscure resolvers that no one had ever found before, or that we didn't know about. Every single person who did this found some surprises. So I thought, oh, shoot, you know. I've got to do something with this because, I mean, basically this meant that there was a huge list of potentially better resolvers than a given user might know about.

So I took the list, we added a bunch that we knew about, and I ended up with 4,854 global resolvers. Because one problem was that the list that the Benchmark has tends to be a little U.S.-centric. I mean, it's Level 3 and OpenDNS and Cox, and it's clearly biased toward the U.S. But I wanted the Benchmark to be useful to people in the U.K. and in Australia, everywhere, globally. So I added a complete new feature to it where, in a separate phase, a user of the Benchmark is able to build their own custom list of resolvers to benchmark out of a master list which GRC maintains. So the GRC server has a master list. Currently it has this 4,854 global resolvers. And you're able to ask the Benchmark to build from that list a custom list for you. It takes a while. It takes, like, 37 minutes because we're literally going out and sending five performance tests to every single one of 4,854 possible name servers.

Leo: Wow. All right. So you got 4,500...

Steve: 4,854.

Leo: Awesome.

Steve: What happens is, when you run the Benchmark the first time, and it uses just the built-in default list of 70-some well-known servers, that'll give you a good sense for how the DNS servers you're currently using, whatever you've got configured, whether it's your ISP's, or your system is using your router, and Lord knows what your router's using, whatever it is, it'll compare that to this well-known list of sort of U.S.-centric resolvers. When it's done, it pops up a notice and says, hey, I notice that you're just sort of using the built-in list, which is fine. But there are resolvers all over the place, some which may be much better for you. The one thing we learned is nothing matters more than distance. So...

Leo: Oh. So you want a resolver that's geographically close to you.

Steve: Exactly.

Leo: Interesting.

Steve: I mean, it really, really matters. And so anyway, this offers you the ability to build a custom list. You only have to do it once, and it does take a while. As I said, it takes about 37 minutes. And, I mean, I've got all kinds of spinning numbers and flashing lights, and so it's entertaining while it's doing it. You get to see the progress as it's going along, and how many of what it's found, and, like, what the fastest and the slowest that it's found so far are. Once it's done, it sorts that entire list of 4,800-plus resolvers and takes the top 50 and builds for you an .ini file, this little initialization file, so that those are - and all the normal resolvers, all the resolvers on the normal master list, they're in there, too. But you get the advantage of just an amazing potential DNS resolver database.

The other thing we do is the top 200 are sent back to GRC anonymously. And that's something for privacy reasons you can suppress, if for some reason you don't want to have anything go back. But it's useful for us because over time I would like to make that build-your-own-list process faster. It annoys me that it's 37 minutes long. You only have to do it once, but still it would be nice if it were a lot quicker. What I'm doing by having the Benchmark send back anonymously the top 200 is I'm counting how many times those resolvers were useful to anybody. Because certainly there are some that are just in this master list that are never useful to anybody.

So after about a year, maybe, I'll take a look at this database that we've accumulated, and I will throw out all the ones that, for example, never even made it into the top 100, or maybe even into the top 50, depending upon, like, how the distribution of them turns out to be. That will allow me to hugely reduce the global resolver list and forevermore speed up the process of people producing this personalized list. But so once this is done, the Benchmark then has a customized list that will be different, I mean, truly different for every single person who uses it. Very different in the U.K. than in Australia, than in Singapore, than in the U.S. And what our users found is surprises. When they then run the Benchmark against that, they find faster resolvers than they ever knew about before. So it really is a learning experience, as well. Some of them you may not want to use. I mean, it might say John's Muffler Shop or something.

Leo: Well, I'm getting, for instance, the top right now is NTT America Technical Operations. That's the Japanese phone company. Now, I don't know if I'd want to use their DNS. Now, Google is still pretty high on the list. But you know who the highest is, Level 3.

Steve: Yeah, Level 3 does perform very well for a lot of people. And again, it's a function of where you are. You may be geographically close to a Level 3 server.

Leo: Right. This is on the EFM network that you're on. In fact, I think I'm breaking you up a little bit just by doing - does this use a lot of bandwidth, or no?

Steve: It was a tradeoff for me. The problem is, you wouldn't want packet collisions to lower the reliability. Nor would you want running a benchmark to, like, get in its own way. So it's self-throttling. There's a lot of time was put into the development of this. But frankly, I'm really proud of it. It's just, for 163K, all assembly language, more than anything I'm happy that it's done. It's got...

Leo: Now, are you allowed to use whatever you want?

Steve: Yeah. Well, that's the other thing. I mean, these are all publicly available DNS servers. So...

Leo: They don't have to make them public.

Steve: They don't, exactly. Anybody who wanted to could easily lock them down so they're only available to smaller networks, or only to their own users. In fact, some people may find, I show red ones where they're just not available at all. So some people will find, for example, that, like, Cox DNS servers are not available to them, whereas they are for other people. And in my case it happens that Cox has a couple of very fast ones near me, even though I'm not a Cox user.

Leo: I'm getting Neustar.

Steve: Ah, that's the name for UltraDNS. That's UltraDNS's new name.

Leo: What is UltraDNS?

Steve: They're a commercial DNS provider.

Leo: Well, they're fast. Wow.

Steve: Yeah.

Leo: Yeah. NTT is still number one. Then Symantec Corporation.

Steve: Yeah, Symantec's server is very fast for me, too. Which is interesting.

Leo: We may be on Level - you're on Level 3. I may be on Level 3, as well. So maybe...

Steve: Actually, I'm not. GRC is, but I'm on Cogent's bandwidth. So, yeah. And people will find that their mileage varies, that it varies from - based on where they are.

Leo: You may prefer OpenDNS for the features it offers. And by the way, OpenDNS is right in there in the top 10, along with Google. And Google I do because I can remember 8.8.8.8 and 8.8.4.4. easily. So I use Google whenever I can't remember anything else. And I use...

Steve: And Google actually is on Level 3's bandwidth.

Leo: Ah, no wonder they're coming in so close to each other.

Steve: Yeah.

Leo: Yeah. And OpenDNS I use because I use it for the filtering for my family. And it's certainly better than my Internet Service Provider, which isn't on the list at all.

Steve: Do you see some little green or blue rings around OpenDNS?

Leo: Let me look. I see green rings around NTT and Level 3. OpenDNS is, no, it's just a neutral beige or orange.

Steve: Okay. The reason I ask is that OpenDNS does have that feature which allows you to do DNS rebinding attack prevention. And if you turn that on, you'll see it. This Benchmark does detect when servers have DNS rebinding protection in them. But what's interesting is it tests both over IPv4 and IPv6. And OpenDNS is only blocking IPv4. And I'm hoping, one of the things that I'm hoping is, as this benchmark becomes more well-known and popular, that it will exert pressure on providers, like OpenDNS, I'm sure they'd like to close down the fact that you can bypass their DNS rebinding protection just by using an IPv6-style query, and you still get back a private IP, which you can't get over IPv4.

Leo: So now I've run it. And it says "Consider creating a custom name server list for yourself."

Steve: Okay. And so if you click that...

Leo: And build custom list. And so it's going to build a list based on the results, the 72 resolvers who have the best performance in what I just ran.

Steve: No. It starts from scratch. It's going to scan all 48 - it just pulled a master list, a current master list from GRC. And it'll now run through all 4,800 of those, showing you also elapsed time and how much time it's anticipating it'll take. So that'll be counting down.

Leo: How big were the differences between the best and the worst in your tests? I mean, is there a significant difference?

Steve: Yeah, really significant. I mean, we don't have time to let this run right now because it'll take about 37 minutes.

Leo: Right.

Steve: But you will find resolvers you never knew about that are, like, around the corner from you somewhere that somebody has open and available.

Leo: Isn't that great. Well, I'm going to - yeah, I won't keep this running because it's also eating your bandwidth, I can tell.

Steve: And if you do click the Conclusions tab, you'll see that from having run the Benchmark, it'll, like, give you a summary in English of, like, exactly what it found.

Leo: Really neat. Really, really a neat project and a fun thing to do.

Steve: Ah, and it's done.

Leo: Well, somebody said that the best part about writing, like for a book, is having written. I imagine it's true for programming, as well. The nice feeling of completion.

Steve: Completion. I'm glad it's done. It's a beautiful resource. It's free for everyone to use. I think people will get a lot of benefit from it. And there's a ton of web pages on GRC to back it up. I've got complete, thorough documentation for the entire thing. So I think it's a nice resource now for the Internet.

Leo: Well, that's great. Well done, Steve. Thank you for nine years in the making. The DNS Benchmark, it's done. It is good. Thank you, Steve.

Steve Gibson is at GRC.com. That's where you'll find 16KB versions of this show, along with the 64KB versions, along with the text versions because he gets the great transcriptions done, along with show notes, along with DNS Benchmark and all his other great stuff. But you know, you can take advantage of Steve and do all those free things. But if you do, it wouldn't be a bad idea to buy a copy of SpinRite, too, because that's the world's best hard drive maintenance utility. It's fantastic. Recovery, too. People often wait till they need it for recovery. Get it now to maintain your hard drive so you never need the recovery part of it.

GRC.com. Follow him on Twitter, @SGgrc. And next week we'll be doing a Q&A session. And I'm sure there are people with questions about this and all the other things we talk about. So if you go to GRC.com/feedback, you can ask your questions there. GRC, Gibson Research Corporation, GRC.com/feedback. And Steve Gibson, thank you for being here. We'll see you next week.

Steve: Oh, we should just mention, too, that you can find the Benchmark just in the menu under - if you go to GRC.com, under Freeware, and then Utilities, and there's the DNS Benchmark, so.

Leo: Very good point. We almost left them to find...

Steve: Easy to find.

Leo: It is easy to find. Hey, you know, we're doing - because I'm going to take the week after Christmas off, and most of our shows are going to go dark, and we were doing "best ofs" for a lot of the shows. In fact, if you go to TWiT.tv/bestof, you can vote for the "best of" for TWiT, for TWiG, MacBreak Weekly, and just something you remember that happened during the year that you liked, anything that happened in 2010. And we were going to do it for Security Now!, and then we realized, wait a minute, we don't need to. There's one episode of Security Now! that stands above all others.

Steve: Oh, I know which one. How fun.

Leo: One episode that should be repeated every Christmastime, every year.

Steve: How fun.

Leo: You know what I'm talking about, the Portable...

Steve: I love it.

Leo: ...Dog Killer. And I want to thank the person who sent me an email saying, well, you're going to rerun that; right? I said, solves that one.

Steve: Perfect.

Leo: So on, I don't know, December whatever that is, I think Christmas is Saturday, so whatever that Wednesday is following Christmas, in between Christmas and New Years, I'm not going to be here, most of our shows are dark, a few won't be. I think Sarah says "I'm doing a show, darn it." So I think, I imagine they will be on. But most of the other shows will be in "best of." And Steve Gibson, everybody can hear the Portable Dog Killer episode. That'll be great. Thanks, Steve. We'll see you next week on Security Now!.

Steve: Thanks, Leo.