LONDON – Tech Day
Monday, June 23, 2014 – 11:00 to 17:00
ICANN – London, England

EBERHARD LISSE: [inaudible] I'm an obstetrician and I am proud to say that I don't have a night job anymore. I'm not an obstetrician anymore. The malpractice insurance is too high, and now I have more time to play with things.

This is now the 24th Tech Day that we're doing, and though I'm very proud of the number, the round number will be in LA, so we're looking for what to organize it together with DNS-OARC in LA. It will be a two-day meeting on Sunday with DNS-OARC topics and on Monday with Tech Day topics, keynotes, and the like.

As usual, we have a little bit of a mixed program. First speaker is Kim Davies, who will talk about label generation rule sets at IANA, then Gavin Brown from CentralNic will talk about real time zone updates for DNSSEC zones.

Is Gavin in the room? There you are. Thank you. I haven't met him, so it's always good to know that my speakers are available.

6114161151611616117161181611916120161261616126161t we did in Singapore and what's happening to our domain, and Nils Clausen from our local university will talk a little bit about research he did into that.

Roy Arends will then give the host presentation and I've asked him to build in a few things about how they do the data analysis on their DNS [side]. Then Paul Hoffman is speaking about DNSharness. That's some tool that they have developed. He said the presentation is going to be

20 minutes. The demonstration is going to be too long for a Tech Day, so he will set up individual meetings for people who want to see this action.

Lunch this time is self-catering as it was in Singapore. There are so many little shops around that I'm not worried that anyone will die of hunger.

In the afternoon, we have an update from the IETF. This is more or less going to become a standing opportunity/occasion. As you all know, we're trying to extend the reach from purely ccNSO into TLD space because as we are feeling that smaller gTLDs are beginning, gTLDs have the same problems in many ways. On the technical side, that's more ccTLDs have had. Then I think Henry Stern from Farsight will talk about stuff they're doing. Then Jaromir Talir will talk about registry object locking in FRED. He pointed out that I should have mentioned the word FRED, but my agenda line length is not long enough for it.

Then we have extension or a follow-up on what we did in Prague, where we had a Name Server Round Table. This time I did not forget to invite one, so I've got all the major name servers, including Microsoft and PowerDNS, who I forgot to invite. PowerDNS I forgot to invite in Prague, so I made sure that this time I didn't.

UNIDENTIFIED MALE: Thank you.

EBERHARD LISSE: You're most welcome. No, I was really upset with myself for doing that, and once I noticed, I told you that you had a standing invitation at any opportunity.

Andrew Sullivan for Dynamic DNS is going to moderate that Round Table, and I thought that one of the major users [inaudible] company should moderate this and he's been very helpful. He's already stirred up the pot a little bit by communicating with the participants so that we go a lively discussion going.

Then we'll have or usual closing remark. I have Norm Ritchie on it, but he may not be available, so [Ondrej] will have to start thinking which one of you is going to do it.

Which means without further ado, Kim Davies is going to be next. We're going to present from the Adobe Connect directly. In other words, the presenting slides are going to go straight on the outside to the remote participants. So we'll see how that works.

Once we're done, we have two room microphones, so I will use one set. [inaudible] will use the other set. So if there are questions, feel lively.

KIM DAVIES: Thanks, Eberhard. My presentation today is on label generation rule sets. Before I get started, I wanted to just talk about the terminology real quick. Many of you will have heard of IDN tables. Historically, for the last ten years or so, ICANN has encouraged those that implement IDNs to document them in what we call an IDN table, to submit it for publication, and that was pretty much it. There was no real sort of restriction on the formatting. It was for IDNs explicitly.

Now we have a term, label generation rule sets, which has emerged in the last year or so. The reason for this is a few factors. One is that I think firstly IDN tables doesn't really convey what they are. Label generation rule sets is more explicitly about how do you generate labels from a given input.

Secondly, it's not actually unique to IDNs. You can have a rule set for an ASCII label, permissible code points or permissible rules related to ASCII generation only. It's certainly not explicit to IDNs.

Thirdly, it's not actually limited to domain names. You can have rules about labels that are used for other purposes altogether. So rather than restricting this concept specifically to IDNs, this new name I think better reflects the notion that this is a method of defining the rules used to generate labels. In the compelling context, it's for IDNs, but it can be certainly used for other purposes.

So without further ado, all right. This project is about coming up with a standardized approach to expressing registration rules for labels. I mentioned that we now use the term label generation rule sets.

Firstly, it should allow all the existing IDN tables that are out there to work, as well as the known registry policies that are out there. Informing this work, we looked at all the IDN tables we had at hand. We looked at all the ones ccTLDs have published to date. We looked at all the proposed tables that were submitted in the new gTLD application round. We sort of did some other registries we were aware of that might not have submitted anything formal to ICANN, and we essentially reviewed all the different registries we could find to look at what rules they were using, and that helped inform defining this format.

Another goal is that it be an objective machine readable format. One of the key reasons for undertaking this work is that, if you work at corpus of IDN tables that are out there right now, some are ASCII, some are HTML, some are PDF. As the recipient of them before they get published, I can tell you we get some in far more exotic formats than that even, and then we usually kick them back.

However, the intent here is to share knowledge, and our view is that the best way to share knowledge is something that can be objectively read, that there's no room for error or miscommunication. So if it's a machine readable format that's objective, that is a key win.

The vision here – and I think this is the most important part – is the idea is we have this LGR file and you can put it in some kind of Runtime that understands this file format. Then, as long as you have an LGR Runtime, then you can slot in or out LGRs based on evolution of registry policy. For backend providers, if you have a new customer that has a new, unique policy, you should be able to simply drop in an LGR and have it work. The idea is you probably can eliminate a lot of hard coding, complex validation rules into your backend. Then I mentioned at the outset this is not specific to IDNs.

So a bit on the why. I told you that we have IDN tables, and it's a bit of a mess. The net result is a lot of registries put the more difficult logic about their registry policies encoded in their back end. It's not something that's published in a file format. So even those registries that publish IDN tables, we know that the table isn't a full story. You can't look at the table in isolation and necessarily know all the registration rules. So it's only half meeting the goal if you know a list of eligible code points, but nothing beyond that.

The second part of the why. Like I said, we maintain this IDN repository. It's a mess, basically. The new gTLD program has struggled with this, too. The new gTLD program sets a certain bar in pre-delegation testing where gTLD operators need to submit their tables and have them reviewed – sort of a sanity check. They've seen as well that there's a lot of inconsistency in the way the data is presented.

There are two RFCs out there. Firstly, RFC 3743 documents the Chinese language approach and has a certain table format that is suitable for their application. Secondly, there's a model format in RFC 4290. Both of these, however, suffer because they don't provide you enough tools to really represent the different registry policies that are out there. So having common format will help us reuse these tables, validate table format, and so on.

Lastly on the why, and the reason why this work has really kicked off in the last year or so in earnest, is that ICANN has an IDN variant project right now. The goal is to come up with rules in the root zone for allowing variant TLDs. The way that's working is the goal is to have sort of a master set of rules for the root zone circled root LGR, and that will be a unison of all different rules for different scripts. So there'll be an Arabic rule set, a Chinese character rule set, and so on. These will be unified in some fashion, and then you'll have a root LGR.

So it's sort of a prerequisite to this work that we need the ability to take these disparate tables, analyze them, merge them, and then form this root LGR.

The net result of this work is an Internet draft (Draft-Davies-IDN Tables). The name of the draft was pre-coming up with this terminology. It's currently in the eighth draft. We're still likely to make modifications to it. Really, my purpose here today is to encourage those with an interest in this topic to look at it, review it, and provide feedback. But I'll get to that in a moment.

So what does this draft describe? It describes a file format. The file format is XML-based, and it's for describing label generation rule sets. Label generation rule sets comprise of allowable code points, contextual rules, dispositions – and what I mean by dispositions is certain variants you would register, but certain others you would block. So it's not just a matter of permutations; you need to derive actions for those permutations.

The co-authors of the draft is myself and Asmus Freytag. Asmus is one of the co-authors of the Unicode standard. To date, we've had strong input from those who have been involved in the IDN Variant Project. The IDN Variant Project has a very specific mindset, I would say, so ideally we would have further review as this draft continues on.

At its most basic, what does it do? You have two pieces. You have an LGR tool, or an engine or whatever you want to call it, that understands LGRs, and then you have the rule set itself. So with those two pieces, you should be able to feed in an input of a domain name, and if you want to a simple validation check, it can tell you, "Is that domain valid for that particular registry? Yes or no?" So that's the basic use case.

A second use case is variant label generation. You input a particular label. Again, it runs through the rule set through a tool. It emits a number of different variations, and then depending on the policy as it's described, some might be allocated, some might be blocked. There's a certain set of verbs that are described, but if you needed to add to that, you can do so as well.

Key here is the concept that they can be merged. The fundamental concept behind IDN tables was reuse. The reason ICANN started its registry or repository of IDN tables was the notion that we don't want everyone reinventing the wheel to reinvent these kinds of concepts, so registries should be able to reuse what's out there and adapt it to their needs. So the notion of merging them, doing other kinds of things, doing comparisons, unions, differences – that kind of thing – should all be possible.

I know based on my research for this work that there's countless, for example, Chinese language tables that have tens of thousands of entries in them. No good way to compare what are the differences. Are they identical? Are they very similar? Are they very different? Intuitively, it's hard to tell. The notion that you can basically just run a simple dif between two tables and it says, "Here are some rules that slightly differ. Here are some code points in one but not in the other," is quite valuable when doing analysis of these tables.

I won't go through this kind of example in detail, but suffice it to say that, in practice, even the tables that are out there today have a lot of complex interdependencies. You have asymmetric relationships between certain code points. In one direction, one would result in the other being registered, but in the other, it wouldn't. So all this is possible with this file format.

What do these LGR files have in them? Firstly, they have code point lists. Generally, that's the fundamental building block of an IDN table. Essentially, what are the code points that are allowed for registration? You can tag code points with tags, and those tags in turn can then be used to inform actions. So you can tag certain code points with certain values, and then in your registry policy, you might handle them in a slightly different way.

It supports variants. Variants can be 0 to n code points. In some cases, you might map a variant to null, so the variant would not have an alternate code point. Typically it's a one-to-one mapping, but it can also be one-to-more-than-one code point.

Variants can also be conditional by meeting certain tests. You could have rules in there that said, "This variant only applies in this situation," and that situation is then described by a rule.

We have whole label variants. So instead of operating at the code point level, they operate on the whole label, kind of like regular expressions. You can say, "This is a variant, but only if this code point follows this class of code points/only if this code point is at the end of a string/beginning of a string," and so on.

The fourth point I think is a big [win] that we'll see the benefit from over time, which is that I think a lot of the IDN tables are out there the author has gone through the exercise of looking at the Unicode standard, applying some filters, and then generating a big long list of code points.

Instead of deriving it from the Unicode standard, why not apply those filters directly, and then as the Unicode standard evolves, the file format of the table doesn't actually have to change? So we can leverage those Unicode properties to directly populate the table.

The meta data is in a standard format, and there's a clear schema, so you can validate tables and so on.

So just the most basic table you can probably devise – this is how you represent an LGR for LDH labels (Letters, Digits, and Hyphens). It's XML-based. It's a little verbose, but beyond that, I think it's relatively straightforward. You have a data element containing a single code point, and then you can have ranges of code points. Right now in conventional IDN tables, you actually list them out all individually. Here you can just define a range.

Here's one example that touches on a few of the more evolved concepts that you can't really have in IDN tables today. This is implementing one of the IDNA context rules, which says that you can only have a zero width joiner when it's following a virama. The way we do this is we say the zero width Jjiner, which is the value 200D is only valid using that when clause when the rule joiner is satisfied.

If we look down at the rule joiner, it's a contextual rule that says if you look behind, there needs to be the value of the class virama. Then if you look at class virama, rather than defining what that is, we actually rely on Unicode properties. So it turns out viramas are defined by a certain property, ccc with a value of nine. You need to look that up.

The point is that, rather than replicating the complexity of the Unicode standard, we can leverage it and make these tables in turn relatively simple.

What's the current status? It is the standard format, even though it's in draft for this IDN variant project. There's a bunch of community generation panels that are developing their rules using this format. There's an integration panel. The integration panel's job is to merge those generation panels' output into that single common root LGR.

As a starting point, the integration panel has just released a document I think earlier this week called the Maximal Starting Repertoire. What this is is basically the base set of code points that would be allowed. Everything that in the future of this process will be a subset of those. So that Maximal Starting Repertoire has been published in this format as well.

Those that have reviewed it seem to think it's a good idea, this work. And no one, at least to my knowledge, has said it's a foolish idea. So thus far, it seems to be the right thing to work on, but we're certainly looking for opinions on how to evolve the work.

What's the long term vision of this work? Some of the, I think, success factors here would be firstly we have this IDN table repository on IANA.org. I think all the existing tables there can be ported to the new format. We can heavily automate that repository as well. Ideally, that repository will be set up XML-files that you can plug and play and uses this format. We can automatically do validation checking and all sorts of stuff that's not possible with the current formats.

Depending on the maturity of this work, it's quite possible that this will be the required format for IDN tables in the next round of gTLD applications. Many of the issues we've seen with the current round hopefully will be addressed by this common format.

I think one of the underlying assumptions here is if it does its job well that it will be in the interest for a registry backend provider to implement an LGR engine and then define registry policies using this format. Then as you add new customers, as you add new registries, you can just slot in new LGRs and just have it work. You don't need to write new code.

So I think those three things are achievable. Whether they happen or not is another question. But I think that's the long-term vision of this work.

What's next? Specification is stabilizing, I think. As we get more eyes on it, we are more and more confident that it does what it needs to do. Right now we're looking more towards completing software implementations. So we have three different software implementations right now that are in various states of maturity. We've done sort of mini-bake-offs on certain features. Ideally we want to have at least several complete implementations to fully test the specification.

We also don't intend to finalize until we have real world experience. One thing we've noticed with these generation panels, some of them have exotic ideas about sort of contextual rules that we might not have thought of, so we want to make sure that it's expressive enough that those rules can be implemented in the standard.

I mentioned in the beginning that we think this has broader utility than just for the IDN variant project, so it would benefit from wider review, which is why I'm here today. A strong ecosystem of implementations would benefit us and we think that there's benefits for registries in doing so.

One open question: right now this is published as an informational Internet draft, but would there be value in moving this to the standards track instead? So that's an open question and I appreciate input. So that's it. Thanks. I'd appreciate folks having a look at it.

EBERHARD LISSE: Thank you very much. My own personal question is that any of this stuff should go into the RFC process, whatever track, and that is because just the way that ICANN sets technical policy, which is not really ICANN's mandate. So if you come up with good stuff, have it reviewed through the normal process. That's my own opinion. Have it reviewed through the normal standard process and have it refined by the people who do this all the time. Then in the end, it gets approved as an RFC, and then it's done. That's my personal opinion.

Any other questions? We've got two microphones, one for the right and one for the left. Okay, Peter?

PETER JANSSEN: Hello. PeterJanssen, .eu registry. First of all, the URL doesn't work, but that's a detail. You are familiar with the German Eszett sharp S where IDNA 2003 specifies that it should be translated to the string SS, where IDNA 2008 specified it as a first class citizen. Some people registries might still consider even under IDNA 2008 rules that the German Eszett sharp S should be blocking the SS string or vice versa. With the current specification that you just presented, with that allow to do that sort of thing? Where one unique code point character blocks or bundles up or whatever to call it a string of Unicode points?

KIM DAVIES: Yeah. Sorry. It should be able to express that kind of logic. Right now the format is agnostic towards IDNA. It's our assumption that separate to the table you'd probably implement IDNA 2008, but there's nothing explicitly implies one or the other. But if you wanted to implement as Eszett handling rule, you should be able to do it using this format.

PETER JANSSEN: Okay, thanks. Second problem, if I may. You briefly mentioned the transitivity or how the non-transitivity of certain rule sets where one implies the other but the other doesn't imply a third one, where the third one doesn't imply the first one again.

This sort of implies a direction to certain rules where one thing coming from one let's call it script to another is a rule, but the other way around is not. Was this something that can be easily expressed? I can't come up with the good example for the moment, but certain characters in one script might sort of want you to – you would want to block something in one script if you come from that script in another script and not the other way around.

KIM DAVIES: I would think you could, but as with many of these things, we get to get examples, and then we can test that assumption and work it through. I think one thing that has come up in the LGR work is some of this symmetry you actually want to test for, so one of the features that these LGR implementations – the tools – will have is sort of sanity check on your table that if you wanted symmetry in your table you can run it through the tool and you can just test for it. Because there is a no one requirement for asymmetry, that's not a stipulation in the format, but it's something that you can test for.

GAVIN BROWN: I have a question, if I may.

EBERHARD LISSE: Just before I forget, everybody taking the microphone please identify yourself to the remote audience who cannot see but only listen.

GAVIN BROWN: It's good to hear that there are software implementations. Are they going to be open source?

KIM DAVIES: I'm hopeful at least one will. To be honest, I have one that I've been writing myself that's on GitHub, but it's actually the least mature and the most out-of-date.

The most mature one we're trying to work out how to open source it. It kind of relies on some closed source stuff, but I think we'll get there. But definitely that's the intention.

EBERHARD LISSE: All right. Good. Next up is Gavin Brown – yeah, give him a hand of course.

[applause]

Next up will be Gavin Brown from CentralNic. We have a little bit of an interruption time between presentations because we have to load them into the Adobe Connect so that the remote participants see the same thing that we're seeing at the same time, which is often a bit of a difficulty.

Was there any question in the chat remotely? Is anybody looking at the chat, other than [Christina] who's busy? I always try to give remote audience precedence with questions and I usually forget that I'm doing that, so if there is remote questions, [Christina], please let us know.

GAVIN BROWN: Okay. So my talk today is about a design of a DNS system that we think may be of interest to this audience. So we're using currently with our gTLD registry environment, which is obviously kind of handling fairly large volumes at the moment because one of our TLDs is growing very quickly.

Before I get into it, however, I just want to just give you a brief history of our DNS infrastructure because it's been around for about 20 years, so it kind of gives you an insight into how the system has evolved over the time.

The first kind of precursor to what became CentralNic was established in 1994. We have very little information from that far back in history, but my understanding is that the registry system ran on an Altos Series 1000 system, probably on Xenix using an Informix databases before it was acquired by IBM. So the registry was pretty much all manually kind of maintained.

When a new zone file was generated, which was probably once every few days, someone would copy the database over UUCP to a SunOS box in a data center somewhere to generate a fresh zone file. Obviously, this was well before any sort of DNSSEC stuff, so you'd have two name servers and a zone file would be generated every day or two and anything that was changed the registries during that time would get pushed out the master server and the secondary.

By the time of about the year 2000, things had moved on a little bit. We were using BIND 8 on Slackware and we could have compiling BIND from source and combining everything else from source because it was Slackware. But essentially the system works more or less as it used to.

By this time I think we had a database server that was close to the name server, so there'd be a program running on the name server that would query the database and generate a zone file. That usually ran I think every three hours. This time, we still only had a handful of DNS servers. I think there was one in London, one in Germany, one in [Telehouse] London, one in our offices – the data center in our basement.

This is pretty much the situation when I started in the company in 2001. In the time between then and now, obviously an awful lot has happened. In 2007, we did our initial DNSSEC deployment. The way we designed our DNSSEC infrastructure was to introduce signer as a bump in the wire, so we put a singing appliance from a third-party vendor between our primary server where the zone files were generated and the DNS server that served zone files to our public service.

This allows to transition quite painlessly because it meant that the unsigned zones would run directly between the self-primary and the distribution secondary, and only the signed zones would go through the signer.

This is the diagram explaining how the DNS infrastructure works now. You can see the previous infrastructure is still here, more or less. We have database, Legacy master server, signing server, distribution, secondary. But we've now added an additional system for our new gTLD registries, and we have built in high availability and a multi-site resilience so that we're not obviously not dependent on a single point of failure. Everything's configured as a logical cluster, so we have a resilient database cluster, in front of which there's a resilient DNSSEC singing infrastructure based on open DNSSEC. We have hardware security modules each site. They synchronize with each other as well. Then we have public distribution secondaries that are [in the mesh] as well.

This was all designed in about 2012 and was deployed either that year or the year after. It is a kind of fairly modern design designed to meet requirements of a gTLD registry.

Before we go further, I'll talk a bit about the signing configuration. So the way that it works is that we have a tool called genzone which writes zone files to disc. It does a big database query, pulls the records out of the database and writes them to a disc. It's very fast and very powerful.

When it finishes generating a zone, it will tell open DNSSEC to sign the zone. It just uses the command line to tell it to sign the zone. Open DNSSEC signs the zone. It will then use the command line tool to tell BIND to reload that zone, and the name server then sends NOTIFY packet to its downstream secondary servers to tell them the new zone is available for transfer.

Last year, we had a business requirement to move to a dynamic DNS update model, so rather than generating a zone every 30 minutes or 40 minutes or however frequently we were doing it, we had to publish zone updates almost in real time. So if the domain name gets registered or modified in some way, then that had to be reflected in the DNS within a matter of seconds rather than a matter of minutes.

The obvious solution for this was the use of RFC 2136 dynamic update. I'm just looking at people on the front row who have done this [inaudible] other registries. So [we knew it would] work.

Obviously that would have to go to the unsigned zone because we don't want to start messing around with the signed zone because we don't want to put the singing part of the system in the application. We want to use the open DNSSEC software to do the signing.

That meant that we have to now provide access to the unsigned zone [over] Port 53 so that dynamic updates could go into the unsigned zone before it gets signed.

We also had additional requirements that we didn't want to spend any effort building new infrastructure – no physical infrastructure and virtual infrastructure. We didn't want to spit out extra virtual machines. We didn't feel it was necessary and justified. And we wanted to do everything on a single system.

The solution we came up with is BIND Views. These may or may not familiar, so I will give a brief explanation of what they are. A view inside BIND is essentially a virtual DNS server, a bit like a virtual host in a web server. You could have two DNS servers that are configured and behave differently, but inside the same process, the routing determining which view inside BIND a query gets delivered to can be configured based on the source or the destination address.

One thing that you can realize about this is that it means you can have the same zone in different views, but with different data sources. So you can use a different zone file. You could have a .tld zone that uses one zone file, and the same TLD in a different view is a different one.

So this is how we've implemented this system. We added additional addresses as [aliases] on the service network adaptor –one for BIND, so it has one for one for another; and one for open DNSSEC because now open DNNSEC and BIND are communicating via NOTIFY and zone transfer.

Open DNNSEC listens on that IP address and when it receives a NOTIFY packet then it does [inaudible] file. Then BIND has two views as I've mentioned before: an unsigned view and a signed view. The unsigned view uses zone files produced by the zone generation tool, but it also accepts dynamic updates from the SRS, and it will then send a NOTIFY packet to the open DNSSEC system. The signed view uses zone files produced by open DNSSEC, sends NOTIFY packets to the slaves.

This is the diagram that shows how it's designed. You can see that it all runs in a single box. As far as the secondary DNS servers are concerned, as far as the applications are concerned, it's just a single machine. What you have is a kind of flow through the system where either full zone updates or dynamic updates go in one side. They pass through the unsigned view into the signer, out the signer into the signed view, and then out to the secondaries.

This is just a quick view of what the configuration looks like in BIND. Obviously this is not complete, but this is kind of indicative of how you might configure it in your DNS server.

We use the match destination rule that allows for remote querying so we can do monitoring and debugging. We could use a match source directive, but that means you have to be – when you're doing debugging, you have to be using the same host or you're forging your packets or something. So it's better to use the match destinations.

Obviously, you've got the configuration to send the NOTIFYs to the right IP address. We also, even though it's not really required because we're doing it over the loopback. We are using TSIG as well to secure everything.

This is the configuration for open DNSSEC. It's just a kind of snippet from the XML file. It's very simple. It just tells open DNSSEC to listen on Port 53 on one of the IP addresses on the server, and when it's finished signing a zone, it uses the RNDC reload zone command, but it specifies the view as part of the command. So it's saying only reload the zone in the signed view because obviously that's the [farthest changed].

This is the DNS adapter file. So again, what we're doing here is configuring the master server, which is the server that we receive NOTIFY packets from and also the servers we do zone transfers from.

This is the Zone List XML, which is the lists of zones that are signers responsible for. A very simple configuration change. You just change the adapter type from file to DNS and then specify the adapter configuration file.

As I mentioned before, it's better to use external IP addresses or externally accessible from your local network so you can debugging and monitoring. We still use the Genzone tool to periodically generate a complete fresh zone. The reason why is that our SRS is configured to send off packets for events that are triggered by registers in our EPP system in our web interface.

If someone performs an action that results in a zone change, that results in a dynamic update going through. But there are some batch processes that run in the background which don't do this dynamic update because we don't really want to be hammering our DNS server with thousands and thousands of update packets when we do a billing run and delete a few thousand domain names, or to spend large numbers of domains oor whatever reason.

So Genzone is still useful. It means that we don't have to run it so often, and it's still key there. One of the things that we did when we were building the system was to test that we had the application code correctly implemented was to use the Genzone tool to generate a full zone, and then using the same database snapshot, try to reconstruct the entire zone file by calling the application code that generates the update packet for every domain name in that zone and just doing a dif of the two to make sure that we're getting it all correct. That was very useful.

One of the things that we do with Genzone is when it does do a full zone update is it has to freeze and thaw – that's the terminology that BIND uses – to suspend dynamic updates as just a precautionary measure to make sure that you don't end up with inconsistencies.

One of the things that we do is we actually query the name server for the serial number. We use that as the basis to work out what the serial number is. If there's a dynamic update coming through at the same time, that that will also result in a serial number increase, and we don't want to be in a situation where we end up with a serial number mismatch where the server thinks that it's got a serial number that's actually lower than what's actually in the zone file.

The RNDC command tool accepts the kind of in, unsigned, in, signed syntax on the command line for most of its functions. So you can specifically freeze the zone in the unsigned view. The in is not in the view. In is the class. It's the Internet class, but it's kind of convenient English language.

We have experienced some issues with the open DNSSEC adapter, but we're getting very great support from the people who are developing that, and we're confident we'll able to resolve the small issues that we have.

We've not nailed down exactly what the problem is, but it could be that it's something related to the serial number race that I mentioned just now.

So I've finished my talk. Questions?

[TRONG]: Hi. This is [Trong] from Neustar. I'm sorry, I might have missed it, but what's the advantage of having both the signed and unsigned zone? Was that for HA or DR purposes?

GAVIN BROWN: I'm not sure I understand.

[TRONG]: You said that, let's say for TLD, you have two views, right? One is singed and one is unsigned.

GAVIN BROWN: That's right.

[TRONG]: What's the reason behind that?

GAVIN BROWN: Because you can't send a dynamic update packet to the signed zone because it will clobber the signature. Only the signer generates the signature. If our application code did the DNSSEC signing itself, then we could update the signed zone and save ourselves a little trouble. But I don't really want to implement a full DNNSEC implementation in PHP. So we haven't done that.

TRONG: Okay.

EBERHARD LISSE: This is from the Chair. I was wondering why do you do this in the first place? Why don't you just use two different hardware?

GAVIN BROWN: We could do. We could have done. It wouldn't have been dramatically—

EBERHARD LISSE: It's an interesting concept. I'm just wondering what was your rationale to do this?

GAVIN BROWN: Just because we had already put the effort into building the infrastructure. We didn't have to go out and do it all again. We could have done. I don't think it would have been any more complicated. Obviously more hardware means more things that can break. I guess if this thing breaks, it will break all at once, but it was just something we thought was quite neat and it helped keep things self-contained.

ALEX MAYRHOFER: Alex Mayrhofer from nic.at. I was actually wondering about the same thing that Eberhard just mentioned. So I understand it's just more convenient for you to keep it on the single machine because we have a similar infrastructure, but it's separated over a couple of virtual machines so that we don't confuse the sounds actually in the same process.

Another thing, do you also have some kind of validation before you actually send out the sound?

GAVIN BROWN: Yes. We have a step. This is the configuration I gave you simplified but obviously here we go through – we use [LDNS] for verify zone [inaudible] and various other tools to make sure we aren't sending something out that's bad.

ALEX MAYRHOFER: Okay. Thanks.

EBERHARD LISSE: Peter? Thank you. Peter?

PETER JANSSEN: Thank you. Peter Janssen, .eu. Have you considered actually to sending dynamic updates to a signed zone and having the name server doing the calculation of signatures on the fly and all that instead of doing the two-step approach where you have an unsigned and a signing and then a signed zone?

GAVIN BROWN: Yeah, we could do that, but that would mean changing the DNS server that we use and changing the way we do the DNSSEC signing. There was this process of small incremental changes rather than a big change where we junked the whole system and built a whole one from scratch. That's the reason why I did the whole history of our DNS infrastructure – to show it's progressively enhancing rather than evolutionary.

PETER JANSSEN: Okay. The reason that I mentioned this is because you said there was business requirement that you wanted updates to live and in a few seconds, but as I see this, the moment that you do a dynamic update, there's this full zone transfer to your signing machine that will do a resigning of the complete zone, which means that there are full zone transfers going out to all your slaves, which obviously takes a lot longer than—

GAVIN BROWN: No, BIND uses incremental zone transfers. There is a full zone transfer between the unsigned view and the signer, but it's happening over a local interface and it's very quick.

From the signed view to the secondaries, it’s incremental.

PETER JANSSEN: Yeah, but if your singing machine resigns a complete zone, that means that the complete zones, so I guess that your hidden signed master, doesn’t have any history about updates whatsoever, so there will be the full zone transfers going out.

GAVIN BROWN: That’s not what I see. The open DNSSEC, when it receives a zone file over a zone transfer, it will compare it to what it has in the signed zone and [inaudible] the differences, plus anything that needs to be [inaudible].

PETER JANSSEN: I agree, but the fact that you have resigned would mean that all signatures would have a new expiry date and all that, which means that practically all of your resource records would be different.

GAVIN BROWN: There are open DNSSEC experts in the room, but my understanding is that it will only sign the changes, plus anything that needs to be signed because it’s about to expire.

PETER JANSSEN: Okay.

GAVIN BROWN: It doesn't to a full resign even though it may have a full new zone. We certainly don't see very large zone transfers. It happens very quickly. We don't see much bandwidth being used. We only see incremental zone transfers.

PETER JANSSEN: Okay. Very cool. Thank you.

[TRONG]: It's Trong from Neustar. Sorry again. Just a follow up to that question. Your Genzone that you have to run periodically the [whole zone], how long does that take and how big is your zone?

GAVIN BROWN: Well, we have many zones, the largest of which it runs at about 60 seconds.

TRONG: How large is that? How many resource records?

GAVIN BROWN: It's 450,000 domains, so I guess if you average that out, it's going to be about three million resource records.

TRONG: Do you use NSEC or NSEC3?

GAVIN BROWN: NSEC3.

TRONG: Okay. Thank you.

EBERHARD LISSE: All right. Thank you very much. Quite good presentation.

[applause]

All right. As I said, I just want to give a quick feedback on what we were doing last time in Singapore. Let's just find the right button to push. We used to look at the text lock files that we were writing and have now looked at some tools. There is DSC, which is much difficult where the collector is very difficult to set up. I found a little tool called DNS Cap, which is quite old. It comes from DNS-OARC. It was written by Paul Vixie and Duane Wessels, which basically can look at the Ethernet and write a binary format, which I don't understand.

So I Googled a little bit around, and I found that [inaudible] – and Patrik Falstrom confirmed it – had a little tool called [inaudible] in the presentation. You can see here in general that's the link if you need to go to either DNS Cap or this one. Just click on the link and it takes you right to the website where you can download it from.

It's basically an SQL frontend into the [inaudible] so I can easily query this. It can output JASON which I don't understand, and CSV and XML. It has a web server, which I don't need. And it's very fast, which is what I wanted.

So, very simple. Compiles on [inaudible] needed to load a library [inaudible] or two. You need to have to compile a running so it's few packets that you need to load. There is no packet for it under [inaudible.] You have to use the [inaudible] install routine, which we're all familiar with. It took about 20 minutes to get this working, most of it in download time.

So this command is a little bit longish, but I don't want to go into the details. I want to arrive at the end of my presentation and show you the pictures, which is more interesting.

Basically you write this command with some parameters. You write a shell script – PCH.SH – here, which will then be executed in this case every 300 seconds, every five minutes. You can make it do whatever it wants.

[inaudible] can read the PCAP file whether it's compressed or whether it's non-compressed. So if you have bigger files, just compress them on the front, and then make PCAP read it so it's not an issue, or you can read it with PCAP first and then compress it. If you want to keep it, you can remove them, whatever.

I then load them into SQL. I don't use the MySQL import command because I needed to apply some possessing removing full stops from the end, and it gives you quad notation, but I wanted to have it in [Intega]. But that's not a problem. We can just do this. It's extremely quick.

This is some figures. [I leave that] for the presentation to download. I have then plugged this into a [demo]. It runs all the time. I've written some commands in Arab that basically query the database, generate pictures, put them ever so often into a directory and this webpage, which I'm not going to show because I've got images in the presentation, updates itself every five minutes. So I can see on my screen at work if something is happening.

What I want to do is – and I've spoken with Paul Vixie already and I will speak with Duane Wessels when I see him – is to find a way of maybe write MySQL or PostgreSQL plugin so that that thing talks directly without having to go through additional programs.

Then of course we need to do more analysis, which is where Nils comes in on his own presentation, so I'm not going to get into this.

So this is what I wanted to show you. This is then what I call some eye candy. It writes every five minutes. It updates. For me that's good enough. I can do it every minute if I wanted to, but for me it's good enough. Every five minutes is okay.

The left one are the queries that go over the wire. The right one are the ones that have all the legitimate names that we have in our database for this particular area that we're looking at subtracted. So this is only the illegitimate queries.

If there is a certain mean exceeded, it changes the color on the website so it attracts my attention visually.

This is then over an hour, 24 hours, so you can see at night the [inaudible] don't work so much. On this image, you see a big spike. A

French company was again trying to enumerate us, so I threatened them again with criminal charges, and two days later, it dropped to zero and they wrote me a written letter, signed and on PDF that they're not going to do it again.

This is what's even cooler. Three lines of codes in [inaudible] present the data of one day in this format. Unfortunately, I wanted to have these two things together, so the image is a little bit small, but on the website, if you click that link on it, it's much better to see.

The interesting thing is if you have all queries, most of them come from the States. Even if I remove the Google web servers 8888 and 88804, the U.S. is still the biggest number queries. Namibia has a few. South Africa and Russia has a lot. China has a lot. But only if you remove all the spam queries, you find that there is no legitimate queries coming from Africa.

In other words, the only queries that you are getting here are from basically viruses. That was an interesting picture for me to see. I never thought about it in that context. But you see really here most of these countries have zero queries is white, and this is 1-15 queries is dark red, so basically we're getting nothing other than a virus from them.

In the meantime since Singapore, we had a big spike at some stage, so I found out it was a local university. Not this one, but the other one. So I communicated with them. They were very pleasant and very helpful and they were very impressed that we could tell them. They were not so impressed they had to clean up 1000 computers to get the virus off. They know which one it was and they figured out where it was and they redirected their name servers relatively quickly, but they were quite impressesed that they had to go and clean 1000 viruses.

We also note queries from another university where we see the internal network, which is IOMNet, and not [iomnet.au.na]. Now I gave it away [inaudible]. It's leaking out. But it's not massive, but we see the queries because I was actually looking for that, and we are talking to them so they will fix it. They just need to block the egress. It's no big deal.

But the point is these graphics took me about half an hour in between two patients when I was bored. Three lines of code. Database query brings about 150 lines back. You connect it with one line to the object [nl] and then you issue a display command and it displays this. This is really cool stuff.

The data comes with [inaudible]. We can get it more refined. We use MAXMIND scale [ep data] to resolve it by country. If you're interested to do it inside countries, you can purchase their – we use the public open source data which is not as [refined]. If you want to go more in more detail you probably have to purchase it. They can go to [inaudible] so specifically pinpoint where they IP address is currently located, which if you want to see where the hotspots are, it's probably interesting.

All right. So without much further ado, I must now find out how I load this. Christina, how to I load the next presentation?

Nils? Nils is a lecturer at International University of Management, and he's done some work about open source business analysis. Somehow we got in touch, so we asked him to put some stuff or some insight into it. So he came up with some stuff.

What you want to do is you want to not only know the queries that are not in your database, but you also want to, for example, look for ones that are virus-generated that only differ by one character and they go through a whole system. You want to do this not manually. You want to have a computer. Tell them "We have a query. We have a query coming from this area." If it comes from many IP addresses, then it's a bit difficult. So you want to ask the out command if possible tell you we've got something going on. Nils is going to explain how to do that now.

NILS CLAUSEN: Sounds good. Thank you. All right. So, yeah, Eberhard, thank you very much for the introduction. The title of my presentation is "Detecting Attacks Utilizing the Levenshtein String Distances." This is something we kind of developed together.

First thing is I'd like to give you a short context statement of the problem, assumptions, and then proposed solution. Context is NA-NIC has turned on the protocol options on their name servers as Eberhard said, and it gets replicated into a relational database. Basically, you can take any database. It doesn't have to be MariaDB. Then the resulting table in a log contains all name server queries with the time stamp, client ip/port and query name. This is the important information at this point in time which we use.

NA-NIC has noticed attacks. I call them suspicious queries on their name servers possibly caused by bots and viruses and perhaps misconfiguration of networks.

There's spikes in query numbers, as Eberhard has pointed out – certain times, certain days, certain timeframes. The point is these attacks are only originating from an easily detectable uniform range of clients, but they can be distributed. There's different character permutation techniques that people seem to use, so we were thinking on how to detect them. Simple substring comparisons are useless.

Our assumptions are as follows. We say that only a small number of queries or suspicious queries are only occurring a small number of times per distinct string, meaning if you're looking for a domain name that's really a legitimate query, you will have more than one or two or three queries in a certain timeframe.

These queries show a somewhat similarity. You don't know in what way. They can be issued from various clients, even at the same time. You have a bot network, they will be fired off at almost the same time. And they sometimes do not necessarily produce the peak in the number of queries, so it's not sure that they really come above a certain threshold.

Query names that exactly match register domains are considered to be legitimate, and while this is something Eberhard did as well by subtracting those illegitimate queries from some originating countries, they're down to zero, which is interesting.

The Levenshtein string distance measure is a metric for measuring the difference between two sequences, and it counts the minimum number of single character edits, meaning that you as a human person, how many type keystrokes would you need to transform one string in the other.

It was invented or formulated by Vladimir Levenshtein in the '60s, and this technique is also used by search engines for suggestions when typing areas are suspected, so meaning, "Did you mean this and that?" There's also a demo on the University of Eindhoven by following this link.

The proposed solution from us is as follows. We take na_log as a basis and we take the attributes – query time stamp, client IP, and query name – and we do pairwise calculation, meaning we take the set of data, duplicate it, and have a matrix in the end. Then we compare everything with everything. This is an exponential rate of growth, which is not really good, actually, but we can still limit those number of combinations.

Then we also derive aggregate attributes like day, month, and year to have somewhat of an analysis capability saying, "Okay, in what timeframe?" comparing this month to this month or this day to this day. You can derive all kinds of aggregate attributes from that, like day of week. You can compile all kinds of stuff like holidays in China to see if there's any pattern.

What you can also do – what I didn't do in this case, but it's basically possible – to do further calculations on the results, meaning correlation metrics or for cluster analysis, seeing if there's anything that is being linked together.

So this is the illustration of the proposed solution. You have the na_log table, which I can show you here. So a little bit bigger. You see here this is the na_log table query year. This is a derived attribute – query month, query date, client IP, net is also derived, ISO, query name. So this is basically the table.

Yeah, we do one-on-one—

EBERHARD LISSE: Carry on [inaudible]

NILS CLAUSEN: Yeah. Cool. At least it goes back to where it was. So we do a one-on-one view and join those both, and then in these cross-sections, you can do the pairwise calculation on the Levenshtein distance.

I will go to these slides in a second. As I said, pairwise comparison implies exponential cardinality, so that might impose pretty long calculation times. So I would suggest if you do this, don't do this on a small [inaudible] database. Do it on an in-memory database or something. Or you can use so-called materialized views or do intermediate tables to do all the joining.

What you get is this. You have basically the client IP, the query string, and then the comparison query string. Of course, you don't compare query strings to each other, so we don't do the cross-section of each other, but all the other ones. And everything that is from a high to a medium similarity. So you don't do things that are very much apart.

You see here there's a permutation – 1CMEQ versus 0G5EQ – and this query was only fired off once, and the Levenshtein distance is three, meaning you need three keystrokes to transform this string into this string. Then I do a Levenshtein ratio calculation, meaning you set this

Levenshtein distance in correlation to the length of the query name. Then you see like 27% of these two strings are apart, but that's a medium similarity. You can easily see by comparing all the strings here – this is a computer program that does that, or somebody very bored. You never know.

EBERHARD LISSE: Or very cheap labor.

NILS CLAUSEN: Very cheap labor. I don't know. What you can then do is you can sort and do all kinds of things. Now at this point comes all these business intelligence measures into place to analyze this stuff. You can even use pivot tables if you like, so there's no real reason to have a big tool set behind it. Or you use [inaudible]to visualize the data.

But then you can see that this IP address here is some kind of a bad boy, and this one as well. Then you can try to find out who that is by contacting the appropriate people or companies.

By showing the range of the Levenshtein ratio, you can easily see that there's pretty uniform distribution. Sometimes there's not so much similarity – 0.28 – but I cut if off at 0.3. I say everything that's less than 30% similar is not similar in my point of view.  You can still change this boundary. It doesn't matter really.

How did we do this? You need to download, first of all, the Levenshtein module and compile it into your database, or if you use [Oracle] for example, you do PL/SQL function and put it in the database. Then you

| | |
|---|---|
| | have a function. In the end, you run those scripts, which I've included in the – well, no. Yeah, I can distribute those as well. Then you can run those queries and create those views and then you get the results set from where you can do  the analysis from. So yeah, that's pretty much it. Any questions? |
| EBERHARD LISSE: | Any questions? Thank you very much in the meantime of course. I've expressed by thanks before. |
| NILS CLAUSEN: | Why thank you. |
| [ARNALDO]: | Hello. Arnaldo [inaudible] from [C Machines]. |
| EBERHARD LISSE: | What's your name? |
| [ARNALDO]: | Arnaldo. |
| EBERHARD LISSE: | Okay. |
| [ARNALDO]: | So you said at some point that you took queries of the same length. Why is that? |

NILS CLAUSEN: Well, the assumption is that the permutation is done with same-length queries, but you can take this off the script and say you want to compare everything with everything, regardless of the length of the string. This is also possible. It was just for me to limit the number of combinations.

[ARNALDO]: Did you try using a metric index to accelerate your experience?

NILS CLAUSEN: You mean on the database?

[ARNALDO]: Yeah, like [M3] or some other type of similarity data structure?

EBERHARD LISSE: So he's running it on our – the database is sitting on one of our servers.

[ARNALDO]: No. It's just to accelerate the process of finding the closest Levenshstein queries to a given query.

EBERHARD LISSE: He just explained. He's running this remotely on our servers, and our server is an old Pentium. It's 2 gigabytes and nothing fancy on it, so we haven't done any hardware – we're looking into investing into hardware, and then we can look at this. We found that if you run more than one day's worth of queries, it takes about six days to analyze, so we stopped looking at big volumes of data because the hardware is just too small and the memory is too small.

This is just a proof of concept of what you can do. We're looking at buying a bigger machine with more memory to run this on appropriate hardware. Then we can look at using a different database to run this in-memory and speed up the tables and see what we fine-tuned this. That's probably what you're referring to.

[ARNALDO]: Thanks.

[MARTIN LEVY]: Martin Levy, CloudFlare. Actually, it's a follow up to your statement. What was the total query size that you were using through some of these numbers that you have on the screen and when you were looking at the IP addresses on one of the other pages there? Can you give us some idea on that?

NILS CLAUSEN: Yeah. We went with closely to 100,000 queries on our hardware. We took 100,000 queries and then exponential. Well, this where the hardware kind of stopped working.

MARTIN LEVY: What was [inaudible]?

NILS CLAUSEN: I think Eberhard can give you a better number for that. I think it's 200,000 per day. Is that true? How many queries do you have per day?

MARTIN LEVY: Sorry. Yeah, I will repeat my question on the mic. Yeah, how many queries where coming into .na total for the day, and then what percentage was your test on? Thank you.

EBERHARD LISSE: My presentation shows we have about 250,000 per day. It's one of our name servers that is not commercially – well, it does not have commercial domains. It has .edu, .alt, .org – non-commercial domains. We want to have a sample that we have close by. The [production names] for .com and for a top-level are sitting on [inaudible] and then they told us how we don't bother about it. We overprovisioned.

We just wanted to also see what's happening and also I'm sure that you can do things. 250,000 [inaudible] per day. If I tried to use more than maybe 1,000,000, my hardware doesn't do it anymore. If I look for four days or five days, then it takes too long to do it, but what applies here is the principle that you generate some ideas what you do it, and then, as I said, we audit the [232 gig box] and we run this on that. We will experiment with PostgreSQL. I find that loading into PostgreSQL is just too slow. We haven't really looked at the analysis, but I will put the same table into PostgreSQL run the same analysis and see whether it speeds up the queries. Sorry, Jay was first. Jay actually raised first.

JAY DALEY: Jay Daley from .nz. Thank you, Nils. I'm sorry, I didn't understand why you chose a 30% cut-off.

NILS CLAUSEN: Well, actually that's one of the assumptions, saying that everything that is above this threshold is not similar anymore. You can as well have a higher threshold, say 0.5 for example. But that means that half of the string is not similar or cannot be produced by simple transformation. But you can play around and see what your distribution is in your data. There's one part. Try to explore what your data is like, and then applying certain measures on how to find out what the bad stuff is. You can as well set this to a higher threshold.

JAY DALEY: So to be clear, then your assumption is that the algorithm the bot is using will not change the permutation too quickly.

NILS CLAUSEN: Yeah, that was my assumption.

JAY DALEY: Thank you.

EBERHARD LISSE: Roy?

ROY ARENDS: Roy Arends, Nominet. In a minute, I'll tell a few stories and stuff that we've done on big data, but one story that I want to tell you about Levenshtein distance: if you combine Levenshtein distance with keyboard distance and if you only look at very popular resolver, like for instance in the U.K., [BT] is a popular resolver, [inaudible] is a popular resolver, and very popular domain names, if you take then the Levenshtein distance of one and calculate keyboard distance, you can see on average that for instance so much percent of the population is left-handed because they are likely to make a mistake with their right hand if they're left-handed, and vice versa.

Additionally what you can do if you look at the circumference or the diameter of the keyboard distance to the key that they were supposed to use, you can see that if the distance is slightly larger, then the site is slightly more male-oriented. As the keyboard distance gets a little bit smaller, it's female-oriented. On a very large data set, this is actually true because if you have an enormous amount of data, like 3 to 5 billion DNS requests today, you can see the difference, because on average males have slightly larger hands than females. Just a small side story Levenshtein distances.

NILS CLAUSEN: That's interesting.

EBERHARD LISSE: And there's slightly different interest in male sites.

ROY ARENDS: I haven't done that test, but I'll take your word for it, Eberhard.

NILS CLAUSEN: Yeah, that's interesting.

BERT HUBERT: Thank you for this. I'm Bert Hubert ofPowerDNS. This is not a Namibian problem. You're lucky enough to only see a very small slice of the problem. We run resolvers. We run authoritative servers, and we've now found that the scale of this kind of attack is so large that some of the larger top-level domains are now actively blocking these queries.

Our aim at PowerDNS is to prevent these queries from leaving our resolver because we want to recognize them before they hit you. This means that we are aiming to attack this kind of stuff in real time, which is a bigger, nastier, and hairier challenge than being able to do six days of analysis in one day because we find that the attack patterns are right now shifting. If you don't do anything, they shift once an hour. If you try to damp the attack, they shift once every ten minutes. So this is really a lot of questions.

Later this afternoon, we have DNS Round Table that starts around 3:00 p.m., I think, where we will also be addressing this issue. I'm sure that quite a number of you are either on the receiving end of these kinds of attacks, or on the sending ends on these kinds of attacks. In both cases, I hope you can join the discussion because we have never seen this level of DNS attack before, and we have now seen .com shut down on us because they had attacks they could not manage. This is something new for us.

EBERHARD LISSE: My point is to show that you don't need big resources to come up with bright ideas, yeah? Once you have the idea, of course if the hardware, as I said, it's a Pentium – 15-year-old hardware. Okay, we put a [Ubuntu 12] on it, but otherwise it's old and it does nothing but DNS and a bit of e-mail, so it's perfectly in order.

But if you want to run queries, you have to really buy a nice box –with [Oracle], as I said, a big 32-gigabytes, 3 gigahertz box with a few costs see what happens. But the principle, the proof of concept, that there's a problem, and how to analyze, I found this very interesting especially since in a small country, we have also local resources, which is important. Many small ccTLDs or gTLDs are overwhelmed by this and they don't know what to do.

If you understand the problem, then it's much easier to come to a solution by talking to the DNS [when does] what to do, how to block it, how to say so. I fully agree with what you're saying.

BERT HUBERT: The data is important, but we will probably be discussing this later on. Thank you.

EBERHARD LISSE: All right. Okay, then I thank myself and Nils very much.

NILS CLAUSEN: Thank you very much.

EBERHARD LISSE: Christina, can you load the presentation? And then Roy Arends or his deputy or both are going to give the host presentation.

UNIDENTIFIED MALE: Who's up next? I thought—

EBERHARD LISSE: Roy Arends and then Paul Hoffman.

UNIDENTIFED MALE: Oh, okay [inaudible]

EBERHARD LISSE: From which I assume you are Paul Hoffman.

UNIDENTIFIED MALE: [inaudible]

EBERHARD LISSE: Okay. Christina is going to bring this up. Roy Arends doesn't need introduction. Adam Leach is his colleague. I know that Roy does very deep data analysis stuff, so I asked him to delve a little bit into this as part of the usual host presentation. Of course, they're not going to give any proprietary stuff away that could in particular – not stuff that can be used to deflect this because this is a public meeting. But I'm quite sure that he's got some interesting stuff to say. There you go.

ADAM LEACH: All right. Hi, good morning. Don't panic. I'm not going to go through that. I just want to say, hello, my name is Adam Leach. I'm the Director of R&D at Nominet. I've been with Nominet about just coming up to twelve months, and this is my first ICANN, so I'm looking forward the rest of this day and the rest of the day getting to know more of you.

So this presentation is going to be a bit like England's journey in the World Cup. I'm only going to be doing the first bit, and then I'm going to stop and then I'm going to hand over to my colleague, who I'm sure a lot of you know, Roy Arends, to finish the job, so to speak.

So I'm going to talk about briefly the U.K. zones, some of the changes we made to our business. Then we're going to talk about Nominet R&D and our focus of our work before I hand over to Roy.

The U.K. zone is about 10.5 million domains, and we established in 1985, with Nominet being established in about 1986. What's behind me here is what was the world's largest welcome sign, and that was laid out underneath one of the flight paths at Heathrow Airport a couple of weeks ago on the 10th of June, and this was to signify one of the biggest changes we had in the U.K. zone – to start accepting registrations at the second level. So we can now register .uk addresses.

On the first 24 hours, we had over 50,000 registrations, which included StephenFriar.uk, [inaudible], Bentley.uk, Burberry.uk, plus many others. It was the fastest -elling domain release on record, beating the first-day sales for the current batch of gTLDs, which is something we were extremely proud of.

Next I wanted to talk just a bit about Nominet R&D for a bit of context. To avoid any kind of copyright infringement, there was lots of fascinating images on Google Images for people doing R&D, but I opted for a picture of our actual R&D team in deep discussion here. Our focus is to build innovatively products and services, which complement our public purpose, which is a key point for us, and create new opportunities for Nominet and its partners.

We have a nice wide remit of topics that we can cover. At the moment, our work is focused on cyber security, Internet things, big data, and of course DNS remains a core focus for us.

We're a smallish team. Our research is informed by our partnerships we have worldwide with universities such as MIT, Cambridge University, and Georgia Teach, as well as more established tech businesses.

We're also starting to engage more with the startup community, and that's not just in the West Coast. London here has its own tech hub known as Tech City in the east of London, and Nominet has just recently opened an R&D office there, so we can maximize on the linking up with that energy and infuse us and behind the startup scene.

The thing for today's talk is big data and analytics and we wanted to share some work that we've done. Actually, this is work that predates me by a longshot and has been pioneered by Roy and others at Nominet for the last couple of years, and we're quite excited to show you this work today.

ROY ARENDS: Thanks, Adam. I'm Roy Arends. I'm Nominet's research fellow. I'm going to skip through the slides really quickly. I have a lot of stuff to tell and only so many minutes, so apologizes to the scribes if we have any. I'm going to go through this fast.

A couple of years ago, we had [the need] to just look at just DNS data. However, this is what we had at the time, and this is DSC – apologies for the typo. I just noticed it (BCS). DSC is Domain Statistics Collector, and it actually does it what it says on the box. It collects domain statistics. For instance, how many queries did you have? Or how many NXDOMAINs did you see over the last hour or so?

If you want to deep analysis – for instance, if you want to understand why you see valleys and why do you see peaks in your data and understand why it was and how, when, etc., etc. – you need to have something a little bit more coherent, a little bit more interactive.

So I'm going to tell a few stories in the minute about stuff that we find in the last couple of years. We found an awful lot of things, but I'll only tell you about five of them. Before I do that, I want to introduce you to – ha, great. Something went wrong. Anyway, it doesn't matter.

So we have a tool we call Bumblebee**.** Bumblebee is our big data solution. Initially, we looked at Cassandra and Hadoop and all of these big data solutions, and indeed it does what it says in the box. They are big data solutions. You get for instance large PCAP data sets. You then canonicalize it. You normalize it. You put it on a large disc in a special system file system. You think about what query you want to ask it to do. Then you make some coffee and you get a result back. It's not exactly what you want and you can do it all over again.

Now, Hadoop is actually great if you want to have large unstructured data set and find correlations and relationships and structure in that. However, DNS is highly structured data, and there's not so much bandwidth in the variation of DNS data, if that makes any sense.

So we set out to build our own thing. That thing is called Bumblebee. Too bad I can't show the very first slide. I'm happy to give a demo to a few interested folks afterwards.

To make a long story short, our tool needed to be high performance. It needed to be high performance because we wanted to do real-time interaction. We basically want to touch the screen. We want to set some bits, clear some bits, see what some resolvers are doing, and just get that high interactivity going.

We also want to have something fun to play with without having to do the learning curve to understand SQL. I am not an SQL guy. I'm not as SQL fan, actually, so I just want to have data to play with. I know a little bit about DNS, so that's why focused this too on DNS, of course.

We created a beautiful GUI. The GUI is using all the modern stuff. It's HTML 5, CSS3, Ajax, Java scripts, and the GUI is actually a simple application on top of a platform. That platform is where the big data resides. Everything is related to the term Bumblebee, so of course we have a queen, we have a hive, we have worker bees, and of course, the data itself we call the honey. Of course, you can only stretch the analogy so far, but it's kind of cute when you talk about this.

The API is very, very simple – incredibly simple. It's restful, so you don't have to keep state on a server. You just ask it questions. You get data back. JASON is a very nice format, very easy to parse. All the modern languages can deal with JASON format.

Additionally we also wanted it to be machine-learning ready. The reason for that is we want to have these big algorithms with fancy German academic names that I have no clue what they really, really do to find structure in our DNS data without having humans to understand what the real correlation is.

Now, in order to do that, you need to have the individual packets. You can't do it just with counters. You need to have the individual packets. So Bumblebee captures each and every individual packets, adds them up to the response, and builds the data set that way.

Now, the first story I want to tell you is about – is it really the first one? Yep. Well, Botnet tracking was the first one I want to show you, but that somehow dropped off the slides. I'll just skip to the next one. This is DNS changer tracking. What you see on the left hand side is a bunch of dots. Every dot is an hour's worth of traffic, and the color of the dot – in this case, green – is the ration between stuff that exists and stuff that doesn't exist. In this case, all these names happen to exist.

Now, this is an IP address of a DNS changer, and the reason that this is an abnormal behavior is all of a sudden this IP address became live, and it's an IP address we haven't seen before. It is also doing a fair amount of [nominal] stuff, stuff that I don't show you on the screen. But one part of it is that, when it starts up, a lot of AWS DNS queries. This actually tells you that the entire DNS changer infrastructure is on AWS DNS on Amazon web services.

What a DNS changer basically does is it's a criminal. It scans the Internet for modems – I'm trying to do modems plural; I have no idea what I'm saying – modems that you can just log into with [adminadmin], and then change your DNS settings to something under their control.

Now, end users are unlikely to see it. How often do you check that? How often does your mom check that? So they can actually hijack you traffic. They will let everything through, except for maybe BigBank.com.uk, or some other financial thing. Bumblebee has helped us to uncover that.

Oh, come on. Yeah I think we need to take this off the Adobe Connect if that's possible. The problem is that Adobe Connect tries to interpret PowerPoints and can't deal with the image size. So, apologies for those remote listeners. We're going to switch to a different version – our own original slides, which I've help build, so I know these are correct. Christina, can we just connect—

UNIDENTIFIED MALE: [inaudible]

ROY ARENDS: Okay. Perfect. Thank you. Hoorah! Oh – no hoorah.

EBERHARD LISSE: It will take a moment. Your computer is too advanced for this.

ROY ARENDS: Okay. Welcome to 2002.

EBERHARD LISSE: [inaudible]

ROY ARENDS: Do you know what's fun? Just before I started to come up here doing this presentation, Simon, our CTO, tells me that the Board is in the room, the CTO is in the room, the COO is in the room, all my colleagues are in the room. So, hello, guys. This is me doing a presentation. Okay, go to slide number ten. Hoorah!

Okay, this is the slide I wanted to show you initially. On your top left, you have colored dots, 21 days of it. So every row is a day's worth, so you have 24 dots and every column presents an hour. So each dot is one hour. The size is the volume. The color is the ratio between stuff that exists and doesn't exist.

You have your standard time [inaudible] on the top right. Everything's completely interactive. If I change one, everything will change on the screen. Everything is also touch-ready. That means that if I run as an IOS, which you can do since it's just Java script and HTML 5, I can interact with it.

It's also what we call subsecond interaction. We had this requirement that all responses from the server should come back with a second. Not everything is within the second because some of this stuff has network latency. For instance, if I would give a demo here, there's quite a lot of data coming back to the system. You would see maybe two seconds.

But the cool thing is, if I switch, for instance, if I want to see all the traffic with the RD [BitSet], everything is updated instantly, and it doesn't matter if I look at 21 days' worth or two years' worth of data. It has everything to do with how we capture data and how we store it.

We also have your standard top ends, most popular names, most popular IP addresses seen within the timeframe. Again, a timeframe can be one second. A timeframe can be two years.

On your bottom right, you see a lot of multicolor dots. That's basically I think 15 seconds' worth of traffic, where each dot is an individual packet. You can color all these packets either by their port number of what bits they have set, or even names, etc., etc.

Underwater, there's a lot of additional stuff that we don't want to confuse the user with initially, but everything is configurable. A lot of other stuff is there's signatures, breakdown of volumes, etc., etc., and everything can be changed to different views if you like. Next slide, please.

Perfect. Botnet tracking. This is what I wanted to start with. There's a botnet that sends a large amount of spam, and it does this by looking up the MX records and bypassing the ISP. Now, if you bypass the ISP, they actually talk directly the authoritative server, and that's us.

So we look for, for instance, MX record requests with the RD bits set and for names that do not exist because all of these spammers, they have lists that are convoluted with a lot of names that do not exist, so you get a lot of [NX] domain responses. There's a lot more in this specific signature of this specific botnet, but this is what we wanted to show you today.

On the right, you see a top 100 of IP addresses for this specific botnet. This stuff is great. We give this stuff to, for instance, people who run anti-spam campaigns who want to incorporate that in their anti-spam lists, etc., etc. For us, this is a byproduct, a waste product, so we just hand over the data, if you will. Next slide, please.

DNS changer – I've been there. Next slide.

Perfect. Okay, who remembers the BIND Packet of Death from 2011? I see one hand – oh, I see a few more hands. Okay. So what happened was Bumblebee found this packet of death because we know exactly how a name server should behave. Now, these name servers are authoritative name servers. They all run BINDS – well, most of them run BINDS – and one thing that you know, or you should know, and that I know is that they should not react to dynamic updates. It's an authoritative server. It's not a primary server hidden behind somewhere. It's a server that's open to the public. It should not respond to dynamic updates.

So we looked at how BIND would behave so we could try to color these things by NXRRSet, which is the typical R code you should not see in the wire.

But it was one. Out of three to five billion packets per day over a period of several months, we saw one packet. That needle in the haystack was one packet that actually caused BINDS to send back an NXRRSet. So we look at the packets, we reconstructed that and made small changes to try to uncover the path inside BIND so that this packet would trigger, and hence we found that all then-current servers of BIND were vulnerable to this packet of death. Next slide, please.

This is something we all know. Resolvers should randomize their port numbers. If they do not, they become predictable. If they are predictable, you can guess them. That's what predictable means. And they can then be spoofed, if you will.

Now, we should have obscured that IP address. We haven't done this. I apologize. On your left, you see nice randomized port numbers. On your right, you see a resolver that does not randomize a port number.

Now, this is still a very large resolver that sends in those few a few hours about 85,000 – this is a random sample I took – queries per second. Sorry, queries in the timeframe; I apologize. But there are still a lot of resolvers in our top 100 that do not randomize their source ports. Even worse, there are few resolvers that have not randomized identifiers, which should be fixed.

Another question that research sometimes gets is that, "Can you show us the uptake of IPv6?" or "Can you show us the uptake of DNSSEC over a period of a couple of years?" So this is about how does [an 89 gigapacket] over a period of two years all over IPV6.

Now, these are not queries for Quad-A records. These are queries coming over the interface that is IPv6. You see a nice uptake of IPv6. Now, I have to admit, if I show you this, you think there's a nice increase in IPv6. However, the entire query load in the last two-and-a-half years has actually increased, so I should have shown you here also compared it to IPv4. However, if I put it one slide, the IPv6 graph would be [inaudible] incredibly. There's not real uptake in IPv6. It's just a little bit.

This is the last one that I want to show you: Cryptolocker. Cryptolocker is a nasty piece of malware. It encrypts stuff on your hard disk and it will hold it ransom for a fee. Cryptolocker uses a domain-generating algorithm, and using Bumblebee we were able to identify this DGA. We could actually crypt-analyze it. This is because we're too dumb to reverse engineer the code. With this DGA, we can now see who's querying what on what day.

Now, this is an arbitrary day, October the 10th, for a single domain name of those 1000 domain names that Cryptolocker would generate for that day, and you see this nice pattern. Now, this is an abnormal pattern. No domain name should follow this pattern. Domain names are normally registered to become popular, and when they do, you see a nice uptake. It's kind of this S form if you will, slowly increase and some marketing around it, and then it [tailors] off towards the ceiling.

This is not that graph. This graph, as you see this graph, something abnormal is going on. Nothing really going on before October the 10th. All of a sudden, October the 10th, a fair amount of queries coming in for it, and October the 11th, everything stops. This is abnormal. Machine learning will show you this, and Bumblebee will show you this, as you can see on the screen.

Now, because we know the DGA, we can now actually go back in time to find the index case – or in popular terms, patient zero, if you will. On September the 5th is when all the anti-virus companies got hold of Cryptolocker malware, and they generated a signature for it. But of course, queries would start long before September the 5th. We could actually go back in time to a few months earlier and we can see the original queries from the original experiments with Cryptolocker DGA.

Now, there's a whole story behind that. I can't go into that because I understand that this is currently under investigation. But this is something that, for instance, Bumblebee would kind of show you immediately if you just pressed the right buttons.

I think that's it. I'd love to do a live demo, but I can't do that now, initially because there's no way to do that in Adobe Connect I thought, and additionally because there is some stuff when you're doing a live demo you can't predict. There's not a demo problem, but sometimes you see queries that you never know what people are looking for online, etc., etc.

Now, if you have any questions for Adam or myself, feel free to ask.

MARTIN LEVY: Adam, Roy, very nice. Martin Levy at CloudFlare. Can you talk about not the tool but the amount of time you spend looking at the data? Then break that down as in just general interest. This is great. This is data porn. I love it. It would keep me busy. But I'd like to know how much time you spend looking at it and find something versus just looking at it. How does then tool then move you to the next stage of cleaning up DNS in some form or other? Thank you.

ROY ARENDS: I'll try to answer this. I'm a little bit biased because I'm a DNS guy and I love this stuff. How much time we look at it? We have another side of the wall at the system administrators. They look at it a lot. They play with it a lot. I'm on various lists where things are discussed if you see this kind of pattern or that kind of pattern before. I literally type a domain name in the tool and I get all kinds of patterns just throwing at me. So this is really seconds' worth of work.

Since I've been using the tool for a while, I know what's normal and what's abnormal. That doesn't help you. If you know a little about DNS and you know what's normal and not, then you can do the same. However, what's far more interesting is to have machine-learning processes on top of this. If you do that, you can have this unsupervised learning process going on. The system throws patterns at you. This can be quite common. This can be your standard anomalies, misconfigurations, etc., etc.

The way we work is if we see something obviously evil going on, we make work of it. We don't stand on a soap box – this is an exception. We don't stand on a soap box and talk about this. You won't see us in the press if we uncover some malware or whatnot. When we found the BIND bug, for instance, we immediately told [ISC]. This is how we roll, basically.

When we see bad domains that are bad IP addresses that do a lot of bad things, we try to enlighten the ISP involved. We had talks with open DNS. We had talks with Google about their [name server]. Of course, we see a lot of backhands from the popular 8.8.8.8. Warren is sitting next to you. He might be able to tell you a little bit about this. I won't challenge Warren to talk in public about this, but yeah, there's a lot of stuff that we do. I hope that enlightens things a bit here. Thanks.

WARREN KUMARI: This is really sexy and I've seen the demo, which is even sexier. But what can people do with this information? If they'd like to go run Bumblebee themselves, is there a way that you're going to make that available? It looks cool. People are going to want it. Now what?

ADAM LEACH: Should I take that one?

ROY ARENDS: Yeah.

ADAM LEACH: We're making Bumblebee commercially available, if that answers your question.

[JOHN]: Hi. My name is John [inaudible] I work for [inaudible] Roy, this is all really good, but could you give an idea about how big your database is and how big the collect nodes need to be in order to gather and process all this data?

ROY ARENDS: Sure. Okay, well, John, as you know, that research is the place where all computers come to die, so we were working with the requirement that it needs to work on off-the-shelf hardware. The hardware that we currently use for demo purposes is over four years old. We have, for the demo, two collector nodes. This is really off-the-shelf hardware. We basically get PCAP next to an authoritative server. It listens to the traffic naturally, [packets] things up, and then sends stuff to the queen.

The disk doesn't have to be big, but if you see an enormous amount of data coming by, you might be a little bit I/O bound. It would be good if you have an SSD, for instance, at hand there.

We are currently going through another iteration of the tool to make this work on virtualized environments, not just the queen, but all the nodes themselves as well.

Initially, the requirement was it needed to scale horizontally. What it means is if you have another name server as another worker bee next to it, but now we've got requirements that we have environments that there's an entire data center with only one worker. We know how to solve that problem. We're just building it as we speak. The team currently at Nominet is building it as we speak.

In terms of system requirements, you can imagine four-year-old off-the-shelf hardware works. In terms of the entire data collection, I think we have nine months' worth of actual live data. I think that's the actual legal requirement – or six months.

We have a set of in total 40 terabytes of disk, and we only have 10 or 16 terabytes covered. This is entire packet stuff, not just counter, but everything up and on.

If you're not interested in for instance the actual data or you can't keep it for legal purposes, you can throw that away and be counterdata, which is also still very, very rich. You can also still interact with that. It's very lively. That is a very, very small subset of the entire node.

[JOHN]: Thanks.

EBERHARD LISSE: Jörg?

JORG SCHWEIGER: Jörg Schweiger with DENIC. Thanks, Roy. I think it was a really brilliant presentation concerning the results, and I really would love to see something like that on our side. I just appreciate what you have been doing, and that has been addressed in a previous question quite a bit.

I'm a bit concerned about data protection, so the question would be, if I want to do the same thing, how much effort have you been spending in provisions just to make sure that the data has been used in the right way and that data protection laws are really kept?

ROY ARENDS: Well, first of all, the disclaimer: I am not a lawyer. What I will say right now is on my own account. Yes, we have those questions as well internally. We have a set of lawyers who actually deal with this. To me, it's not interesting at all.

However, yes, we have built in limits to this system. For instance, you can automatically only sort of last n months or n weeks if you're only legally allowed to do that, etc. Yeah, I like to defer that to the lawyers, basically.

ADAM LEACH: It's not a huge burden on our time, so it's just about the amount of time we store data for. But other than that, it doesn't impinge on our day-to-day of using the tool, especially if we're using it internally to sort of look for our own registry.

ROY ARENDS: Just a small more point in that, that's another reason why we don't show live data. We never, ever, ever actually share the actual data that we see. We can derive information from it and share that, but only if the end user and the end system's obscured, etc., etc. So we never share any data, per se.

EBERHARD LISSE: All right. Thank you very much. Very impressive. Okay, Paul Hoffman is now going to speak about DNSharness. We're a little bit delayed, but I think the cooler presentations are less worry.

PAUL HOFFMAN: I'm not going to have as many graphics as they are. Do you want me to compress?

EBERHARD LISSE: Just carry on.

PAUL HOFFMAN: Okay.

EBERHARD LISSE: We're not going to interfere with presentations. By the way, I've done a headcount. I've stopped at 150. We've usually got 80, so I must say I'm very much impressed with the attendance.

PAUL HOFFMAN: We're good. Thank you.

EBERHARD LISSE: This thing is just as slow for Apples.

PAUL HOFFMAN: Greetings. I realize I am between you and lunch, so I might compress a bit anyways. I'm Paul Hoffman. I actually am the director of the VPN Consortium. This is work that I did under contract to Verisign, specifically Verisign Labs. Verisign Labs tries to get DNS research out to the public, and this is one of the things they've done. It's called DNSharness. So I'm going to give a brief presentation today. At the end, there's contact information, and also I'll be giving a similar but not at all identical presentation on Wednesday in the DNSSEC Workshop because one of the things DNSHhrness is good for is researching that.

What's the best way to go forwards?

UNIDENTIFIED MALE: The mouse.

PAUL HOFFMAN: Ah, got it. Okay, I got it. I got it. Next page. That's not the next page.

EBERHARD LISSE: [inaudible] it will come back. [inaudible] maximize the screen there. There you go.

PAUL HOFFMAN: Good. So in one slide, what DNSharness is, it's a test harness specifically for if you have a set of queries, whether they're good or bad queries, that you want to send to a lot of servers, either authoritative servers or recursive servers, this will help you.

One of the things we started with is, when someone comes to like the IETF and says, "Oh, you can't do that, it will break BIND 9.1 through BIND 9.3," or something like that, there was really no way to actually test that. So this is a tool kit that will let you test any kind of query you put out to a huge variety of servers.

It comes with pretty much all the open source servers available, and it's pretty much meant for DNS researchers. The tool is not really meant for "I'm just playing around with the DNS," but if you're a DNS researcher, especially one who cares about how servers respond, this is for you.

You can make the queries or the responders do whatever you want, and you'll see that in future slide. So if what you really are testing is how will servers respond to a broken query or a specific DNS-protected query, things like that, it's good for you. But also, if you have a whole bunch of clients and you want to see how they will all act with server, you can automate that as well.

The other big advantage that we feel for the harness is that if you have done a bit of research and you say, "Doing this I find this," you can publish your research really easily so someone can do a small change to your research and do something else, this is a way of sharing the research.

That's interesting – ignore the top part.

Essentially what DNSharness is it's a large Python script that's running on a LINUX box, and all the servers that you're talking to can either be off-box, but all the open source servers are running on virtual box. So because they all live in one VM, each one has its own directory. You can say, "Start up BIND 821. Shut it down. Start up BIND 822. Shut it down." That all happens for you automatically. You'll see an example of that.

You can have as many projects as you want, and a project is essentially the combination of a set of queries and all the servers you want to go to. Again, it's very important. You can actually test things off of the harness. For example, you'll see when you do queries for example to recursives, you can send the same queries not only to the open source recursives, you can send it off to 8.8.8.8 and any of the other public ones as well.

This is what's included: pretty much everything we could build, which is not all version of BIND 10. Various things would not build on various boxes, but we've pretty much got all of these available. One of the things that's important is we do actually run a lot of versions of BIND 8 because one of the things that a lot of researchers have found is that queries that go to BIND 8 get very different responses than queries that go to BIND 9, and this is a way for you to be able to find that out.

All of the responses can be tracked. You can be doing Wireshark. You can look for outputs such like that. But basically, all of the open source is available.

Let me describe what a project directory looks like. There's a short project description file, which is in JASON, and then there is a program that runs on the harness itself, which is going to be sending off the client queries. Then there's a program that sets up essentially all the servers you want to run.

For example, you might want to do a test where you have configured all the version of BIND 9, NSD and such to do DNSSEC with a certain set of root keys that are different from what they normally do and you want to see how that's going to affect things. You can do that in your project. You just set up the program to do that. DNSharness comes with a bunch of examples. They're all just written in Python. You can even write them in another language if you want. It's done very simply.

This is an example of what your project description file would look like. You have a name, a comment, and you say what your targets are. In this case, the targets are going to be running on the open source VM, and it's running BIND 96, BIND 84, NSD. Again, this is something where for any of the open source that comes in the box, you can specify them with [inaudible].

You can other open source to the box if you want. We would love you to contribute that back. When DNSharness starts up, it tries to grab every available known servers. It doesn't come with all of them built in. It actually FTPs them all in, which means over time it gets some more. Some of them will build, some of them won't. But if someone comes up with something that is another authoritative server, for example, that could go in here as well.

This is an example of what your project file would like for recurser, and what I'm showing here is that in fact it's not just open source that you can be running. If you've got a Windows server 2003 box that is running as an authoritative, you can put that into the pile as well. Just give it its own IP address, etc. You can run Google DNS – however you want to do that.

The program that runs on the LINUX host is responsible for startup. This piece of software started up with a specific configuration, let it run until the query is come in and gone out, and then shut it down in anticipation for the next one.

So you can do more complex things such as, if you want to test – again, for DNSSEC – what happens immediately after you've done a re-key, you can put that into it as well. Then the queries can come from things like just a dig. You can put together a dig. getdns, which I know has been presented here at previous Tech Days, which is the new API. You can write your client with getdns, etc.

I just actually covered that. It's actually reasonably fast. We're sort of used to it taking a while to start up a server and such like that, but if you have a real small zone, which is usually what you're testing, I tested stuff on a really slow laptop and I sent out queries to 268 authoritative servers, which is pretty much all the ones in the harness, and it took three minutes to run. It's really not a big deal.

So this is really good for researchers who say, "Let's try this experiment…Oh, that didn't do what I wanted." You can still run it over the entire set of known open source servers, or external ones, and change and things like that. So that's good for experimentation.

Same is true for when you go out and do recursives, even though of course now you have Internet connectivity issues. There are things in the harness where you can actually put a [NAT] in the way of a query so that if what you're really testing is, "Do certain [NATs] break things? Do certain home gateways break things?" It's easy to set it up so that those will be in the way for your queries and then look at the responses.

As I was saying, you can make broken queries and broken responses, so there's a zone builder that is part of this that is similar to something that – Roy gave me a demo of his last year, and it did some of the things I wanted. I wanted to do something different. But it's easy for you to create zones that in fact you can set this up so that a query is going to get a known broken response, and it's broken in many different ways. It could have random stuff in the answer. It can have multiple answers that shouldn't be there.

You can also use this to build somewhat broken queries – legal queries that are maybe not expected by the servers, such as a DNS query that has two questions and already comes with an answer – things like that. So this zone builder is part of DNSharness. You can use it for fuzzing. It actually has random generation in it. That's really useful, both on sending out queries and on responses. So that's part of the harness.

The harness comes with the getdns API that I talked about earlier and that some of you who've come earlier know about this. getdns is a DNS API that was designed pretty much as a new API for the DNS. Verisign Labs on a different project is funding the implementation of the getdns. So that's especially good for DNSSEC testing, but it's also good for looking at everything in a full response without having to break up the packet yourself. So the harness comes with that partially because they're both funded by Verisign Labs.

Here's my last slide, which is it's been released for a while. We've been getting some good input. You can pull down DNSharness from DNSharness.org, which is a site that is run by Verisign Labs. I thank them for having funded it because in fact this is really useful for everybody, not just them.

If you want a demo, contact me. There's my e-mail address there. I'm around all this week. I have a box set up running in my lab that I can show you examples of and such like that. I'm not going to do a live demo, because basically it's very uninteresting to watch a live demo. You just see the lines going by saying, "Yes, I tested this sever. Yes, I tested this sever."

However, if you have some things you want to see, let me know. I'm happy to grab some time and talk with you about it. So I think I have not delayed your lunch by much. And if there are questions?

EBERHARD LISSE: Thank you very much. Thank you very much. Any questions?

PAUL HOFFMAN: Okay. And if not, like I say, do grab me. I'm happy during the week send me some e-mail or if you see me in the halls, I'm happy to give a demo.

EBERHARD LISSE: All right. Thank you very – there is one question over there. Cristian Hesselman in the back.

CRISTIAN HESSELMAN: Cristian Hesselman. I'm with SIDN, .nl registry. It's very interesting what you present here, Paul. Maybe we should have a chat afterwards because we developed a similar test bed last year, which the DNS Workbench. It also has like all these faulty zones in it and you can test against those, so perhaps there is some form of collaboration that we can engage in.

PAUL HOFFMAN: Absolutely. The back ends for both the how we make queries and how we do responses are completely open, so in fact if someone doesn't like the way I do those, they can use the DNS Workbench very easily.

CRISTIAN HESSLEMAN: Okay, cool.

EBERHARD LISSE: All right, Good. We will meet at five minutes to 2:00 because I would like to start at 2:00. The next presenter will be from the IETF. Is he here?

PAUL HOFFMAN: He is.

EBERHARD LISSE: There you are. Sorry for not recognizing you, but we haven't met before. All right, have lunch.

PAUL HOFFMAN: Have lunch.

[applause]

[break]

EBERHARD LISSE: Usually we must punish the ones that are there for the ones that are not there. That's the way this goes, usually.

Next will be Stephen Farrell from the IETF, and he is going to talk about pervasive monitoring. As I said, we're sort of starting to make this [inaudible] that the IETF will have somebody every meeting to talk about topics that they're interested in at the moment.

STEPHEN FARRELL: Okay. I'm Steve Farrell from Trinity College Dublin.

EBERHARD LISSE: Click there. Not on this one. Click after the next page. It's not quite at this. It needs a proper click.

STEPHEN FARRELL: It needs a proper click? It need a bit of scrolling? It'll do.

So I'm not the IETF. There's a whole of bunch of people here who go to the IETF and just I'm doing one of the security area directors at the moment. So if you're interesting in other IETF things, by all means feel free to use me as a way of talking to other people, or if you're interested in some of the security area things, just grab me. I'll be here until the end of tomorrow.

I guess talking about pervasive monitoring is something that we've been doing in the IETF a good bit for the last year. Basically, the proposition is that it seems that what's been going on the report of actions of NSA and their partners, whether that's nation states or some corporates, coerced or not, is kind of a multi-faceted form of attack on the network, or is indistinguishable from that. If you're a router somewhere, you can't really distinguish the motivations or the actors involved during this kind of monitoring.

What's been reported is not unique. It's very likely that others have been doing it. It's even more likely that now that there's so much public about this that many more people will start doing it, perhaps not at the same grandiose scale, but nonetheless this kind of attack is probably likely to become more common.

I think the pervasive monitoring attack itself, the scale I think is really what's new about the threat model, perhaps. We haven't really seen any new magic forms of attack. It’s just the scale, the breadth of applicability. Also I think we’ve seen that the attacker in this case is willing to use not just a classic passive attack or a man in the middle kind of active attack, but all sorts of combinations. We probably haven’t fully considered this kind of scale of attack in protocol, implementation, or deployments.

In addressing this or trying to address this, we need to realize that a purely technical response is not going to solve the problem. We’ll come back to that at the end. But I think in any case we should treat an attack like we always do. We should try to figure out technical means that are valid. Do I mitigate the attack? Mitigating doesn’t necessarily mean fully solving it. There are other layers of [the stack] that are needed.

So this slide has a bit of a definition of pervasive monitoring. You can read it. This is quoted from RFC 7258, which is a recent PCP that basically the IETF has agreed to.

The meat of what it says is that in doing protocol development work in the IETF, the best current practice is to consider pervasive monitoring as an attack and to try to mitigate it. So that doesn’t mean that everything will be always encrypted everywhere, but it does mean I think that people doing work on protocols that are open to the IETF need to consider this attack and need, for example, to have a reasonable answer if they’re asked, “How did you consider this attack and what’s the right thing to do for your particular protocol?” That will vary by protocol. That RFC you can read. It’s about two pages, so it’s an easy one to read.

The reaction within the IETF to this over the last year. In Berlin in summer of last year in July, there was some other news about [inaudible] was coming out. We had a side meeting about that. In November then in Vancouver, we had kind of a major tech plenary discussion of this, the video which you can still look at. Practically 12,000 watched the video of that for some mad reason. They were obviously quite bored.

But what's good about that is there were 100 or more people in a room of 1000 people, which was quite full, commented on this. We had a bunch of mailing list discussion there afterwards. I'll talk a little bit about it in a minute.

We had a workshop before the IETF in March here in this hotel. The workshop wasn't here, but the IETF was, where along with the WC3 – and it was IAB sponsored as well. We were considering in a bit more detail than we had done in Vancouver what we could do about it, and there's a workshop report that you can look at that has a whole list of things that would be relevant for IETF and WC3 to take action on or think about.

Then also in the last IETF there was a topic at many meetings and there was a few [inaudible] about it. Hopefully from Toronto in July we start to see some results as new kind of specifications come out. I'll talk briefly about a couple of those.

But this is not just an idea thing. There's a more broad reaction in the technical community in terms of the deployments that you've seen, which is kind of similar to what's going on in the IETF and in other organizations.

I guess one of the reasons I came on today is to find out is there a similar kind of thing that's going on here or should be going on here.

What's the kind of work that we've done lately on this pervasive monitoring stuff? Back in December last year we formed a working group called UTA, which is Using TLS for Applications. Essentially over the years we've had a whole bunch of applications or protocols or applications that use TLS and the recommendations for cipher suites being used and so on. [inaudible] specifications that have been written over the years, so the task here mainly is to kind of update those and to come up with some better recommendations that the set of cipher suites that the people should be using nowadays, preferring [inaudible] secrecy. [inaudible] is much more acceptable now than it was ten years ago, and so on. So that work is progressing. As we've come up with new best firm practices for ciphers, TLS essentially cipher suites to use in various cases.

I mentioned RFC 7258 already. I almost call it [inaudible] sometime because it's like 2.5 pages of non-boilerplate and there was well over 1000 e-mails involved in getting that couple pages of text agreed.

So we had kind of a major debate about this and the end result was that we didn't seem to have consensus that pervasive monitoring is an attack the IETF should be addressing. Then this will inform other work that's happening in working groups and IETF and so on, and it kind of sets the scene for that work, really.

We had a couple of proposals in March. DNS privacy, which I'll talk about in bit more detail because it's probably the more interesting one for this community. But also we had one on TCP encryption, so this having TLS-like functionality in TCP. That can be in the [kernel]. So tcpcrypt is one of the proposals for that. So that will almost certainly think be a working group that meets at the next IETF. The working group charter is just being agreed now.

Actually, I have to put my hand up and say a couple of years ago, the tcpcrypt people came along and proposed doing work on this, and I was one of the people against this because we had TLS. So I think it was a mistake, essentially. We probably I guess see better now that we want more defense and depth and a little bit more variation in the kind of infrastructure. And there are some people who like to have TLS-like features but cannot afford to do it. [inaudible] has to be in the kernel. So that should be interesting work.

We also have a group of people we’re trying to get together to go back over old RFCs and do privacy reviews of those to see what things might need fixing. That doesn’t necessarily mean the RFC was broken or that people didn’t care about privacy, but it could well be that the usage models – and again, if you think of DNS, the way we use DNS now and the set of things we do with it, some of them are much more privacy-sensitive than was envisaged when the DNS protocols were being developed. So going back over that, if we can get a team of people to kind of start using old RFCs it would be useful. If you have some time to do that, it’s not hard. Just send me a mail. I’ll send you the links.

Then the Internet Architecture Board are kind of refactoring their security and privacy programs in the light of everything that’s happened in the last year, and we should see more activity from that soon-ish. And

I guess Andrew Sullivan is presumably here somewhere, or will be. He's on the IAB, so you can hassle him or Marc Blanchet.

Just a couple of other things. The TLS Working Group are developing TLS 1.3. The current one is 1.2. This is essentially aiming at better performance and better security properties for the handshake protocol and learning from the slew of other TLS issues that have arisen over the last decade-and-a-half, essentially as we've really deployed this very widely, compared to the kind of cryptographic protocols that we hadn't used a decade ago.

The HTTP Working Group are developing HTTP Version 2 based on SPDY. HTTP 2 does not mean everything will be encrypted, but some of the browser implementers have stated that they'll only be emitting HTTP 2 over TLS. It looks also like that that working group have agreed to define a way as an experimental RFC how you can transport HTTP [URAL] requested content over TLS also. So not just HTTPS, but also for HTTP [URLs], which I think should be interesting and could accelerate the amount of the web that's run over [inaudible] that might have some interesting properties. That's IETF, so join in if you care.

So if you care a bit more about [inaudible], DNS privacy was in March in this hotel. I was going to steal the slides from their mostly, but actually I didn't in the end, so you can ignore that post. There's a mailing list. You can sign up to it. There's a couple of drafts worth mentioning, but these are all unofficial because it's not a working group yet. It's just some discussion.

[Stephan] [inaudible] has a draft on the problem statement, which is pretty good. It sets out the issues that might be worth addressing. And

Phil [inaudible] has written quite a good requirements draft, and then Phil's way of reinventing the world, which is what he does – he sometimes does it very well – but the requirements draft is worth a look. It's a reasonable read, trying to set out a set of requirements that you might want to meet if providing privacy features in DNS.

So what's the problem? Again, this is probably from [Stephan's] draft pretty much. If you look at the DNS queries, the timing, and the content, essentially is the kind of meta-data that might be of interest to people doing pervasive monitoring.

For example, I haven't seen a paper on this but I'm sure you can easily imagine one, that if you correlate the set of DNS queries that you see at some point  in the DNS infrastructure, you could probably fingerprint the webpage that those are being caused by. Whether or not some content author might be colluding to make that more easy, or whether a content author is trying to not make this easy, my speculation is that those kinds of derivations of information from DNS queries are probably quite feasible.

It would be really interesting to get information from this community to back that up or to scope what is actually seen and what can you infer from what you see that might be privacy sensitive. So that's an area where I think this community could really give us some good input to IETF protocol development work to constrain it and to make it much more practical.

The QUEUE NAME itself obviously can be sensitive. It could be political party, some country code, or some ailments, some sexually-transmitted disease or something, and the full QUEUE NAME it seems is sent too often and too far, so it can [percolate] right away to the roof. It seems that arguably there's no need for that. As you'll see, there are some cases where there's people that are using that, so it may not be as simple as just don't do it. But at least in principle you could send the QUEUE NAME much less far and much less often perhaps.

So those essentially are the problems that this mailing list is looking at. Of course, there are problems with the problem, a few of them. One for example is an interaction between the DNS protocol and TLS. With TLS you have the server name indication, which is the way that you would disambiguate which virtual host you wanted – an Apache install, or a the way somebody did node balancing or did some kind of router that's in front of a web infrastructure would differentiate.

You will see the argument that we've seen already that some people say, "There's no point in protecting the DNS if you're going to give away the DNS name you're after in the [SNI] string in TLS." It's a circular argument because you hear the DNS people say there's no point because the TLS would expose it, and you hear the TLS people say there's no point because the DNS would expose it.

So that's not an easy problem. The DNS Working Group are looking at an issue to see is there a way that a client could [update] protecting the [SNI] information or not send it as much and have that work and maybe for some load balancing cases you need to send something. So that's a problem.

The queue names are used I guess if they're sending queries. I don't know much about that, but I guess for kind of valid networking reasons, they're used sometimes beyond where you might absolutely think you have to send them. It could be that there's some interaction here, which in that problem and the issue of the public suffix list, which is actually knowing when you're talking to a public kind of entity in the DNS.

There are proposals I guess. People are looking at for how to map the public suffix list into the DNS in various ways. So if there was a solution found to that, maybe that would help with data minimizations for queue names.

People will also say that privacy was never a requirement. We have DNSSEC and that's not necessarily ubiquitous already, so don't make it harder. I think that's a fair argument. I think privacy solutions will probably be complementary to independent of DNSSEC. So arguably it shouldn't slow it down, but I think it something to be cognizant of. We don't want to start making it harder to do DNSSEC, or because of DNSSEC make it harder to provide some better privacy. So that's another issue to tackle.

The patterns of interaction between stubs and recursive resolvers and recursive versus authoritatives are pretty different. We may as it turns out need different solutions to protect privacy. The privacy problem is probably different in those two contexts, and perhaps sufficiently so that technical solutions to improving privacy will be different. That might be just fine.

People will also say if you get your DNS from the DHTP, then you're providing privacy IN DNS, which somebody can just fake to DHTP. That's a valid point in some context, but not in all. Also, it would turn what's a passive attack perhaps into a more active attack, and I think that's one of the generic patterns that we can see that is to help tackle this pervasive monitoring attack.

If we can take what was easy to do as a passive attack and make it into an active attack, perhaps the active attacks can be detected more often. Being detected is a bad thing for people who want to do the pervasive monitoring. So even if you only provide protection against passive attacks, there can be benefit in doing that. You have to look at the tradeoffs in each case.

Also people might say to you, "We need to keep using UDP. There's no way you can get this done over UDP." Maybe there is. So the solutions I'm not going into detail because no detail has really been discussed in detail. But I think the solution has a lot in common with tackling this pervasive monitoring problem in other spaces. One is data minimization. So don't send as much data that might allow people to do pervasive monitoring. That actually might have all sorts of other interesting side effects.

For example, earlier we were looking at the distinguishing between real and non-real queries. If you send [less] information, you might make that distinction more crisp and easy for example. So there's a lot of interactions here with data minimization and things we need to learn about.

The moving parts. We're doing some crypto to protect privacy better. They're all pretty obvious. There was a couple presentations back at the [bath]. The bits of crypto you need? Very straightforward. We have a lot of people who understand that. How to arrange those moving parts into something that is deployable is a challenge. I think there's no getting away from that, and it's not necessarily at all obvious. There's a good bit of work to do to figure out how to do that. But the bits of crypto you need? That's pretty straightforward.

So the state of play with this is the mailing list we're discussing. You can sign up and talk about it there. Brian Haberman is working with the DNS Ops Chair Tim Wicinski – somebody noted I probably mispronounced it – on maybe fleshing out a proposal for a charter. I'm not quite sure in what timeframe they think that might go ahead, but the discussion on that will be ongoing.

Like I said, you can get a bit more general and say in a lot of the protocols that we're looking at, whether it's the web or DNS or whatever, the mitigation seems to involve some crypto and involve some data minimization.

The crypto we know a good bit about. We learned a lot about how to deploy that compared to ten years ago. The data minimization on making protocols harder to do traffic analysis. It's something we don't know so much about in the IETF. Perhaps in the operator community or in communities like this, you know a lot more, and we need that help. So if you can help and are willing, that'd be great.

I guess also you include registries. I guess WHOIS has never been controversial, I presume, right? So if WHOIS is involved in collecting a lot of data, how that might or might not interact with the current pervasive monitoring may be worth thinking about. I don't know if the [inaudible] Working Group and the IETF are thinking about that particularly. Maybe you could argue that it's a different community, so you're not actually pervasive monitoring because it's just domain owners or maybe [the alternative]. I'm not sure.

But I think considering the pervasive monitoring attack is something I think is well worthwhile doing and factoring this into how you develop policies and how you do operations and so on.

So what we do in general? Turning on the crypto is good. There's a lot of tools available for this at various layers. They're not all as easy as they should be to deploy and I think that's another lesson the IETF has learned or is learning. We need to focus on making stuff easier to deploy as opposed to gold-plating the crypto. I think that lesson is learned.

There's new work, future tools that I mentioned earlier as privacy, TCP, Crypt, possibly [inaudible]. There might be some benefit in doing some kind of opportunistic security there. We have [inaudible] whether it will go. We'll see.

Measuring and gamifying what people are doing seems to work. If you look, for example, at e-mail between mail servers using the [inaudible]TLS and deployment of that seems to have gone from somewhere in the 20-30% of mail to over 50% according to report on figures from Google and Facebook, and possibly more.

It's increasing. It is possible to get significant changes in the usage of some of these crypto tools as people just go and turn them on. If you talk to the people who've done this, the performance impact and so on sometimes is negligible. Sometimes it's real. Sometimes it's just easy.

Measuring and gamifying this is useful. So the XMPP community also has done some of this, which is kind of beneficial to show people how good is the product they're going to get, how good is the service that they're going to use, and so on.

Like I said, with data minimizations and methods to make traffic analysis harder, we have more to learn.

We could work on better implementations. In our deployments, we can turn on features that will help privacy by thinking about it some more. Eventually – we're not at the point yet, but eventually this may start to interact with people's business models in different ways. That will create new business models for other people I think, which is always the case.

For users, target diversity in another generic thing we can do – not having everybody use the same product or the same service. It does help them to some extent in this.

Nearly finally, one thing to do – and again, I've given a talk like this to various different communities, and some of them are more receptive than others – but I think it's definitely true that talking about the issue openly in whatever forum is relative to you is a good thing to do.

Some people do have an issue with that. I think it seems to help some of those people to say, "We're not talking about the motivation of NSA. We're talking about attacks on a network that could be the same attack done by whoever your favorite bad guy is, whether that's a nation state, and if that technology works for large nation states, smaller ones will do it, down to the botnets and then down the individual script [kiddies] and so on."

So if you feel like agitating, go and agitate. But I think more than that is being a responsible engineer is we need to take account of these kind of things and take the broader implications of our engineering and research and so on.

Concluding, the IETF has consensus that pervasive monitoring is an attack. You can read that two-page RFC that described that. We're working on that problem. We would benefit from getting lots of good input from people who have seen the operations part of these kinds of things, which includes yourselves.

I think we all need to consider how we can work to mitigate this. Basically, from at a very high-level view, when and if various societies decide that they think that as a society they also think that this kind of monitoring is bad, the onus really is one us to have the tools available that they can use to affect that decision that society is going to make. I think that's it. That is it.

EBERHARD LISSE: All right. Thank you very much. Any questions?

STEPHEN FARRELL: Just lunch.

EBERHARD LISSE: No. No questions? All right. I can release you. Thank you very much.

STEPHEN FARRELL: Okay. Thank you.

[applause]

EBERHARD LISSE: The real big issue that I'm having is a tendency of governments that are doing it. Companies are going to start doing it. Bad guys are going to do it. Individuals are going to do it. It becomes so common that later as I said script [kiddies] can do it. It's just going to go to increase the cost of doing business for everybody. So it's good from my view that thought is being put into it on a higher level as far as the different technologies is concerned to mitigate it. Thank you very much.

Okay next one is Henry Stern from Farsight. Have I said something which I shouldn't have? Oh, you did that.

UNIDENTIFIED MALE: Magic.

EBERHARD LISSE: Now we figured out how to do deal with this laptop remotely. Even better. All right. Go ahead.

HERNY STERN: Thank you very much.

EBERHARD LISSE: [inaudible]

HENRY STERN: Thank you very much for having my. My name is Henry Stern, and I'm a member of the Anti-Abuse community. So I'm unfamiliar with the audience here, so thanking you very much for welcoming me.

My past history has been with the Apache Software Foundation for the Spam Assassin project and through the Messaging Anti-Abuse Working Group, of which I'm the Technical Co-Chair. I'll be speaking about DNS and effectively your registries from the context of somebody who's trying to do the right thing. So please don't take any of the stones I'm about to throw personally.

What's going on here?

EBERHARD LISSE: [inaudible]

HERNY STERN: I see. That'll do. So just in the context of how domain name registration works from the perspective of a cyber-criminal who's doing their thing to impact the Internet, they'll typically register a bunch of domain names, get them set up on their hosting providers, whether that be on their DNS servers or on their web servers. They will propagate them in some way through spam or through exploitation of websites to put I-frames onto pages, or through command and control channels on botnets. They deliver their payloads, which might be spam or an exploit.

Then we'll come around and we'll block it. We'll either take down the site or we'll push it out on a black list, or we'll add it to an antivirus signature.

The way the cycle actually works in practice is that it's very quick, and they'll tend to be many domain names by the same attacker at various points in this life cycle at any given moment.

Just an example of how we see these in the field. This is just an everyday spam. It took me all of five seconds to find it in my inbox. This domain was about two minutes old when we found it. As you can see in our passive DNS database, it had been hosted on a number of different places throughout that couple of minutes and in the time following it, and it had a name server that [inaudible] was static.

But one of Farsight's main products is a tool called DNS DB. It's a passive DNS database. We collect large numbers of DNS response packets going from authoritative servers to recursive resolvers and index that in a way that security researchers can use. If we were to [drill] down on that, we'd find that on the same day on that same name server, they had used 91 different almost nearly identical names, all of which only for a few moments at very high volumes.

We did some more detailed statistics on this as well over about three months. We plugged in a novelty feature into Spam Assassin on a number of written domains and [inaudible] out the hit rates of those rules.

You'll see for the past month or so nearly 60% of all spam messages, including spam messages that didn't even have a link in them at all had a link to a domain name that was less than 24 hours old. The majority of those were even fresher than that, breaking then down just by the different buckets that I had used in Spam Assassin to do this. The majority of them you can see are at 30 minutes. Normalizing it even more so, so that we have the probability distribution that it would fall into any one of those buckets, you can see that the chance is nearly 95% that it's going to fall in 30 minutes or less from registration time to attack to the end of the attack. So it's a very, very fast cycle.

In the ten-minute bucket, we found that just 10% of overall spam messages all had domain names under ten minutes old. This is a problem because this is also the area where people's spam filters are most likely to cause a missed spam and allow a piece of malware or some spam content to get through.

We found that just by throwing away every new domain, whether we took any sort of evaluation of it or not, it increased the catch rate of actual spam on the test systems by about 20%. So, major pain point.

What's going on is that criminals are able to move so quickly that we just can't catch up with an RBL, caused by a number of different problems including time-to-lives on NX domains and propagation and discovery.

We might look at zone file access as being a solution to this problem because it's an authoritative list of every delegation that was in the zone at a given time. It's really useful because we at least have some sort of ground truth as to what was in the zone file yesterday, but you only get the delegation points for the gTLDs, and really I don't know if there are any ccTLDs that are offered at all to the public. But they're generally only available once a day. So we had this cycle of ten minutes. A daily feed really doesn't do the security domain name a whole lot of good for defending against those attacks.

To fill this gap, we're using passive DNS. As I mentioned, we have a tool that's called Farsight DNS DB. It's a database. We know of about 350 million domain names that we've seen since 2010 when we started collecting, of which there are seven billion unique host names and significantly more unique resource records that have been served with those owners.

In that database, we're doing a several stage reduction that you'll probably see me talk about at another tech conference where I have a slightly longer slot, but we do a successive data reduction on this to cut an input of about four gigapackets per day down to an output of 50,000 domain names for the first sightings that we see.

We publish that into a DNS BL, which is, for anybody who's unfamiliar, it's a domain name block list. It can be looked up by a mail server if you wanted to do a right hand side so you could throw out all domains that come from those, or if you use tool that does content protection like Spam Assassin, you can use it to discard messages that contain hyperlinks or references to those domain names.

We're also offering this as a response policy zone and the five-second version of that is it's a DNS firewall. You can use it to modify what responses a BIND will give back to your customers. So we're offering a market-based solution to this problem, which is not really ideal.

We're also sort of in the prototype phase we're seeing for anybody who might be interested in playing with this, but we're also looking at producing zone files as well, or something that's just kind of like a zone file for when they're not available, again because there's a market need for this. We really prefer not to be the one to actually provide this.

This is really not a good solution for anybody. Like I said, we're offering up a list of new domain registrations and people are using it. They're chucking out everything that's new for periods up to a day, depending on their preference. What that's saying is that the domain names are hemming around. They're being abused much too quickly, whereas legitimate stuff – like Facebook is years old, and that's where most of the traffic goes. So nobody's really going to care that much that they're throwing away a new legitimate domain for a couple hours.

Again, it's really not ideal. It slows things down. We've been working for about ten years to make it easier for people to register domain names and quicker and make the Internet move faster. This is really a gigantic step back.

Some accountability up front would really help this at a lower cost to everybody. Myself is included in this. The fact that there's a market need for this solution is an embarrassment for the DNS industry and the security industry. It says that we can't keep up fast enough and the DNS industry can't keep bad guys away from registering names effectively enough.

What I'd like for everybody here to start thinking about, and I'm certainly not going to tell the way – I'm not trying to dictate the way that your technical policies are driven here, but what I'd really like to see from the registries is a lot more cooperation with us and the security community because we all have common goals. Work on improving on improving accountability for registries of new domains, deal with fraudulent credit card signups in a more effective manner, provide better WHOIS information, provide better identity control over who's actually registering these names.

We also need you to act like the gTLDs and please start on a scalable basis for the number of members of the community. Please consider starting to offer zone file access in the same way that the gTLDs do so that we can all have some ground truth to work from. It will at least have us better than nothing.

Also, please consider providing deltas. IXFRs from name servers would be a lovely way to do it. The Verisign back in the early days of Rapid Zone, which I haven't seen since because they make you pay for it now, but they would offer over FTP deltas of the names. Anything quick for every TLD. If we had this, the security community would be able to keep up.

Something I didn't describe a little bit earlier is that a lot of those domain names get registered about three hours before they ever go live so that we can see them. So there's these three-hour windows of darkness where the only people who know the existence of the domain name are the registries, the registrars, and the criminals, and not the community who's about to be victimized by these. So this would cut this window down significantly and make it much easier for us all to defend ourselves.

Also, especially now that ICANN is offering so many different TLDs, takedown procedures are going to become a lot more critical. Right now, it's really a niche industry – the domain name takedown process. If you can collaborate with one another on an effective way of managing takedowns from the very large number of security vendors and practitioners and the increasingly-large number of registries, that will make all of our lives much easier and make our process of dealing with the root significantly cheaper.

If this problem continues as it is, which I fully expect it will, we might ask you to perhaps take a step back to the way we used to be and slow down pushing out new domain names. Or at the very least, limiting the opportunities for criminals to impact us by limiting NX record changes. Limit the number of delegations that can be changed and offer an exception process for those who have a legitimate need for it. All of these things will dramatically help all of us cut down on abuse.

I hope I haven't overstepped my time by too much, but I'd like to open this up for questions. Thank you very much for listening to me.

EBERHARD LISSE: Okay, thank you very much. The problem with talking to ccTLDs is there are 253 – call it n – of which m to the square of seven different ideas about what's right, what's wrong, or whether they should care, or whether they can care, or whether they know where they are or what.

We had this discussion what was it, seven, eight years in Wellington where a German registrar – I won't mention which one – and said, "Why can't African registries send us the invoices properly?" I don't understand why it cannot be done, but that's the fact of life. We don't even know. We have a working group dealing with contact information because we don't even know how to reach the management of 253.

For example, maybe I live south of Angola. I wanted to register my last name in Angola. It's just not possible. They don't answer their e-mail. It's just not possible. It's difficult. The more responsible ones are getting zone file access to my name server. That's not a problem. If I trust you, we'll make a plan. I'll e-mail you [and Paul] and we'll sort this out. That's not the problem. But we are so expensive in any case that nobody registers with us for these purposes. That's another way of dealing with it.

I like the idea about limiting the name server changes in an automated manner. I'm just saying that you can say, okay, you limit the number of name server changes per domain with EPP. If they want to do it, they must log in manually. That will turn some people off from doing it.

How many legitimate domain names change their name servers more than once a day? Okay, so let's say twice a day. But how many do this on a regular basis. How many registrars? And you will be able to identify registrars doing this because criminals don't go to different registrars all the time. They look at the least compliant one, most pliable ones that are most interested in getting a few dollars for not looking too much, and you will find whose registrar changed name servers on domain names more often. I would be happy to have one domain name registration per day, but that's another thing.

GEOFF HUSTON: Geoff Huston, APNIC. The problem is that the amount of spam in my inbox hasn't changed over 20 years. That's just the base observation about how effective this is.

Then the kind of question is, why hasn't it changed? Because we've constructed impediment after impediment after impediment after barrier. In essence, what we've done is made life more difficult for ourselves, but there's always more vulnerabilities, and as you sort of push on one side, it just reappears somewhere else.

I see this as another incremental step in making life harder for the rest of us. And okay, you stop that particular behavior, but you're not stopping spam.

Then comes the issue if the real issue is actually about spam, why do we look at, if you will, the symptoms of the day rather than the underlying issues? Because this is just incrementalism that ends up making our own systems more complicated: additional rules, additional processes, additional costs for consumers. But the amount of spam in my inbox is unchanged. So this is cost without benefit. I'm not winning here. As a consumer, I'm just not winning.

I understand your problem, but this more about you've got a hammer and you're defining the problem as a nail. But you hit that nail down, another one pops up.

HENRY STERN: Well, this is absolutely true. We've been playing Whack-A-Mole ever since the botnet, and even before the botnet. That's certainly not going to change any time soon. It's a fundamental weakness in the protocols that we use for distributing—

EBERHARD LISSE: Can you speak into the microphone, please?

HENRY STERN: Yeah. It's a fundamental problem that we have with the way that we disseminate information for messaging. But you chose an interesting phrasing here. You said that the amount of spam in your inbox hasn't really increased in 20 years, but if you look at the amount of spam that they're trying to send, that's gone up exponentially. It goes up doubles or more every year. If you look at SenderBase.org you can see Cisco's accounting for that.

So yes, we are whacking away at the moles, but we are at least making some headway. Really, it's a market solution to the much more complicated problem, and we're going to continue to have to do things like mine publically available information to try to solve this problem.

When better collaboration between security community and the registries and the users, while expensive, is significantly cheaper than not doing anything and will I think make us probably better off and less costly in the long run.

EBERHARD LISSE: Warren?

WARREN KUMARI: Warren Kumari, Google. Two questions now. The first is, surely all that this is going to do is make the bad guys register domains and then wait 24 hours a week, a year, or whatever you set the thing to. Domain names are free for bad guys.

HENRY STERN: They are, effectively.

WARREN KUMARI: Okay.

HERNY STERN: And there are exhibits. We call them snowshoe spammers. There are plenty of people who already do that. But if they want to wait a domain to use their domain name, good. That gives us time to find them. They have to host them somewhere, and it’s either going to be in a place that will take down the name before it can do too much damage, or it’s going to be in a place that we can detect.

WARREN KUMARI: But if they haven’t been used, you don’t know that they’re malicious. You register the domain, you let it sit for 48 hours, you maybe put up a picture of a cat because the Internet’s full of cats, and then you use it.

HENRY STERN: Building infrastructure is hard. Yes, you can put it on a site that displays cats for that first day and then move it over to something malicious, but the primary use case for passive DNS, for instance, is to look at the neighborhood that something's in. You couldn't expect somebody to set up a reliable server every day in a new location and use it just that once for that one attack and then ten minutes use another one.

WARREN KUMARI: Well, also you don't see it in passive DNS until the guy uses it, so…

HENRY STERN: Well, you don't see it in passive DNS, but you see it zone file access, which is again one of my asks of the ccTLD community: you might want to please, please do the same thing that gTLDs do because we can find them in that way and we can at least do something about it before the attack [inaudible]. So I would it if they would all wait a day. That'd be great.

WARREN KUMARI: Okay. My second question was, users like to build a registered domain name and use it immediately, so do you think it's realistic to ask people to make their customers unhappy by registering and then waiting before it's delegated? Do you think that's actually going to fly?

HENRY STERN: No. I think it's a terrible idea, but we have fairly limited options. We have fairly limited options in this regard. Like I said, I don't particularly want to do that. I think it's a bad solution to the problem because we've been gaining in ten years trying to fix that.

NORM RITCHIE: Okay. It's Norm Ritchie from Secure Domain Foundation. First of all, thank you very much for your presentation, Henry. I appreciate that. The Secure Domain Foundation was formed exactly to address the issue that you're talking about. It's a collaboration between registries, registrars, DNS in battling abusive registrations.

We already have a bunch of members. I'll offer to work with you to approach them and see what the right solution is for this. This may not be the right solution, but I think there's some other ideas that might help. So have that chat, and there's probably about 60 or 70 people there already who've already taken that first step and agreed to collaborate and we'll work together with them.

HENRY STERN: I'd be delighted. Thank you, Norm.

EBERHARD LISSE: All right. Any more questions? Just before I'm walking back, what Warren says and we see this all the time. They squeal like nobody's business when we don't register them fast enough. We have an automated system so it usually is immediate, but let's say it takes more than five minutes. Then it's a big drama. But the website comes only after two, three, four, five weeks later. So the name server comes up, too.

It's just an example, but criminals go to the .mls, the .whanot where you can get a large volume with EPP without paying anything, use it once and throw it away. We have to sort of not develop small measures, but develop large-scale measures where you're not going into detail. I don't know – I have no answer either – but I must say I see less spam than I used to see before, but I lose more false positives. I have to look in my spam folder anyway because I have to trace where is this bloody message that I was told I got. So I don't know what to do, either. All right.

HENRY STERN: Just one last thing to close on that. It's [inaudible] thing you brought up. This is why the market solution that we put forward is actually working, because, yes, they complain the domain name isn't up, but it does take them quite some time to actually use that website for the first time.

So what we see in e-mail is that all of the new domains that you see in e-mail are evil because people don't have their business process that quick. All being maybe a gross exaggeration, but the mass majority of them, within ten minutes within first use, are terrible domains.

EBERHARD LISSE: Okay. Geoff? I think Geoff is the last question.

GEOFF HUSTON: It's not really a question. It's actually a comment back on this. Part of this is you won't stop criminal behavior, right, per se? That's an evil motive and that will always exist. You're saying we can stop this particular mode of behavior and exploitation. I note in a subtly different area, which is in the area of exploitation of software, there is a stock of zero day exploits out there somewhere, and they're just sitting there waiting for the right price in order to be released onto an unsuspecting market.

If you sort of go, “Well, new domains are evil,” it’s easily circumvented if your motives are bad. Part of this is you need to look beyond what folk do today into what are they trying to achieve and trying to get that.

And is it a case of trying to stop it or make it more apparent? Because trying to stop it hasn’t worked for ten years, and it’s kind of, “Well, that’s a really hard problem.” Is that a silver bullet out there somewhere and we just haven’t really found it, or is this whack-a-mole game unending? In which case, is there another way to identify the behavior and the mode as distinct from whacking each mole of the day? Thank you.

HENRY STERN: Thank you, and you’re absolutely right. When I got zone file access, that was my big hammer for me. It took some of the spammers maybe a week to come up with faster ways of going registration to hosting, so that is what we will see.

EBERHARD LISSE: It slows down the process at the moment. When [inaudible] came, it took a while to figure it out, so blocked it, they come up with something else. But it’s a never-ending story, but I fully agree we must look at it from a technical perspective. How do we win an engagement? We need to win the war. I also have no idea to do that.

All right. Warren has the idea. He will present it in LA.

UNIDENTIFIED FEMALE: [inaudible]

EBERHARD LISSE: Do you work for the NSA now, Warren?

HENRY STERN: Is this a feature that we can amend in Android?

EBERHARD LISSE: All right. I think we're now reaching the point where we have our round tables, so all the participants that are in the room, please [inaudible] on the rostrum. We're having a bit of a problem. Sorry, Jaromir Talir first. Sorry, I forgot. Jaromir, please.

Sorry, I didn't look at the agenda. Jaromir will first make his presentation, and then we will do the roundable. But you can stay in front if you like. But you can make your presentation without slides anyway.

JAROMIR TALIR: But with slides it will be much better.

EBERHARD LISSE: Something is happening in the background there. A mistake. It was actually click.

JAROMIR TALIR: Good afternoon. My name is Jaromir Talir. I work for CZ.NIC. In CZNIC, I'm responsible for running registry and also developing registry. Today I will talk about registry object locking. I'm not used to using the mic.

EBERHARD LISSE: Actually, click.

JAROMIR TALIR: Okay, thank you. You might actually wonder why to speak again about registry OARC because for somebody it can maybe be quite a boring feature. However, it seems to me that this feature is gaining momentum over last year, and maybe the reason is that the domain hijacking is still an issue for domain registries.

Actually, a year ago there was a presentation in Durban I think made by Google about domain hijacking, so you can find a lot of examples in that presentation. The person from the Google actually advised to use more [inaudible] features in registries.

In the [autumn], there was survey among European ccTLDs, and surprisingly, just one third of these of these registries has this feature implemented.

One reason for this presentation is maybe to give inspiration for how this future can be implemented. In the first part of the presentation, I will describe how registry object locking is done in our registry software called FRED, but this is quite old stuff.

But there are two new things that I want to mention also. The first is that we have a new registrant interface, which can be used for domain locking, that we released in September last year. We also implemented a new way how to do administrative locking, which is the locking of domains that is done by registry itself. If we need to lock the domain not on behalf of registrant, but maybe because of some legal issues and things like this. So this is the content of my presentation. [inaudible].

What is registry lock? Simply, recapitulation. It's a sort of secretive feature, a sort of protection against EPP changes of object in registry that are issued by a registrar.

Normally, all changes on objects in registry are done by registrars through EPP protocol. However, if some registrants want to protect themselves for the case of registrar systems being hacked, for example, then they have to request this registry lock via a different channel than this EPP. The registrant must ask directly registry to implement this lock. So this protection is set by registry after appropriate authorization of the request made by the registrant. So [inaudible] registry register locking.

What is FRED? FRED is our open source registry software solution that we develop and use in CZ.NIC since 2007. There are many countries currently using our system in many different ways. They took it as is. Some of them modified it for their needs.

There is also a newcomer in this group. Macedonia decided to use this system since January this year, so now it's about – two, four, six, seven – eight countries with Czech Republic included.

We have some plans for new things in FRED for next month, which includes for example new WHOIS [inaudible] protocol that [you] will release in August. However, today I want to speak about the features that already are implemented in the current version that is available on the website.

How registry object locking is implemented in FRED? At the entry point to this feature is a simple web form that can be and it is for CZ.NIC integrated into registry website. If people or a registry that took FRED, they usually customize it somehow into their own website. The requester on the form usually fills what kind of changes they want to block. Then they fill object handle and the type of object and the means of authentication of requester.

This is actually how it looks like on our website. It's quite simple. It's protected by captcha. It's quite easy.

Going into detail of all these rows in the form, we have two levels of protection. We can either block only the transfer to other registrars, which causes a situation when the registrant believes in his registrar but wants somehow to prevent accidental change of the registrar. And the second form of protection is the registrant can ask for blocking all changes of the object. So not only transfer, but also changes of registrant name servers and all things inside the domain, let's say.

We provide this feature not just for domain, but for all objects in the registry. In our data model of registry, we have four main objects: domain, contact which can be used in the relationship to other objects like registrants, admin, contact, tech, contact; and we have two more objects call name server set and key sets, which is a collection of name server information and DNS key information. So any of these four objects can be individually blocked for transfer or for any kind of changes.

Regarding authorization of a request, each object has so-called owner for domain. It's a registrant and administrative contact. For contact, it's definitely contact itself. For name server set and key set, it's technical contact.

During authorization, the requester must prove that he is who he is and that he has relationship to requested object. We have two means of authentication. One is a letter with notarized signature, so requester print a letter and go to a notary that will put a stamp with information from his ID card, and then we can verify the content according this information that he really is the person that he says he is.

The second means of authentication is using e-mail with digital signature based on certificates issued by official CA authorities in our country.

How it works? After submitting request, requester provides authentication to our clients and to operator. The client center operator will verify his authentication and confirms that it's okay by setting okay [flag in our] administration interface.

So you can see that this is quite a manual procedure that the client center operator must verify authentication of the request. But still, we put the service free of charge. As a result, registrar will start to receive for those operations that are protected response code objects that was [inaudible] operation.

Also, anybody can see the state of the object in the public WHOIS. This is like this until, again, the requester will do reverse operations. So if they ask for an unlocking, again, they have to authorize himself in the same way.

This is a sort of statistics how popular this service is over the last year. You can see that it was not too popular. It was used for just high profile domains. However, it's quite attractive right now because there is almost 1000 requests in the current year, and we are right in the middle.

The reason why it is is maybe the surveys that we released last year – and this is the second topic I want to talk about – and we call this service domain browser. It's a new registrant interface into the registry. There's a URL, which is in Czech, for this service. It's a sort of integration of registry features, our identity service called [MyID]. The [MyID] is something like internal registrar only for contacts. We built an identity provider over the registry. The users or contacts may decide to move the contacts into this identity provider, and all those contacts are quite strongly validated. We provide to those contacts some web access to web portal with web authentication, which includes strong authentication using SSL certificate or two-factor authentication. So it's quite protected.

When we have users or registrants that have high-level validation, we can provide them more services over registry. Up to now, we decided to provide them a cross-registrar view of their own objects, like domains, name server set, key sets. We provide them direct access to [alt info] for code if they want to transfer domains from one registrar to other. We provide them possibility to merge the same contacts into one so they can see all the domains that previously were linked to more contacts, which were the same. Definitely we provide them also the feature of registry object locking and unlocking.

This a screenshot how it looks like in reality. If the registrant logs into the system, they see all his objects – domain names, service sets, key set – and my contact detail, and they can select what domains want to be locked or unlocked and just do it in one click.

With this feature seems to me a little bit resembles how credit card locking works. The banks provide a feature for locking credit cards, and if you want to go shopping, you have to first unlock your card, and after you do shopping and paying, you again lock your card. So with this tool, if you want, you can easily lock your domain for changes and unlock it only if you do some changes via registrar, for example.

The bad thing is that this tool is quite right now not part of FRED registry package yet because it requires the link to that identity provider service, and the identity provider is quite tied to our environment as it has a lot of hard-coded parameters. But we are still thinking about how to make this tool available also to other registries who want to use FRED.

The last thing I want to talk about is administrative locking because every registry must somehow cooperate with law enforcement agencies, and we have to cope with the request from courts or police. We have to suspend domain or block domain for certain kinds of changes.

We also implemented a new feature that anybody actually can ask for a temporary lock for some domain if they, let’s say, start some dispute in some court about the domain and they don’t want to wait until the court will ask us to block the domain. So they can go directly to us with the proper paper and we will set a temporary lock for this domain.

In the past, we used to do this activity because it was quite certain we did that manually by some command [inaudible] tools that was used only by administrators. Because the requester number is growing, we had to integrate it in to the web administration interface. So a new feature of one of the recent versions of FRED is that enhanced web administration interface containing possibilities to do this administrative locking.

The properties of this administrative locking is similar that it’s possible to block many or all of EPP commands, like transfer, update, delete, renew. We set appropriate statuses, like server transfer prohibited, server update prohibited. Together, we also set a new status: server blocked. Again, registrar will start to receive information that objects status prohibit operation.

As part of this administrative locking, we can put the domain out of the zone and activate it because sometimes it’s the content of the records that we have to disable this domain.

As I said, the locking can be bound into some time period. For example, for the request by a person, I think we put the domain  into lock state for four months according to our registration rules.

Again, the statistics. This is the numbers of these administrative locks asked by law enforcement agencies. We are in the middle of the year and we almost have the same numbers as last year.

So the conclusion for my presentation is that even in [three R model], there are many use cases for registry and registrant direct communication like this, like registry object locking, and this can be enhanced by some nice interface for registrants. There is not only voluntary locking requested by registrant, but registry must also provide for their [staff] administrative locking features because this is requested by a lot of law enforcement agencies. Both of these features are currently supported by our open source registry, FRED.

Thank you for your attention. I'm happy to answer questions.

EBERHARD LISSE: Thank you very much. I have got one question. I know the Germans have digital signature tied to their ID card. Tie a card reader to it and then you can sign legal documents in a legal manner. Does something like this exist in Czech Republic yet?

JAROMIR TALIR: No. In Czech Republic, we are quite behind the e-government activities around Europe, so we don't have e-IDs or things like this.

EBERHARD LISSE: Because if a domain is registered with the e-ID documentation, then you can verify it's the owner fully automatic and that can be done fully automatic.

All right. Warren has a question.

WARREN KUMARI: Warren Kumari, Google. Actually, more of a comment. This is ridiculously useful. Since we've had people who've been enabling registry locks, we've noticed a noticeable decrease in hijackings. So this does actually work really, really well and I encourage others to do it, too.

EBERHARD LISSE: Diego from that side? Only one mic, so I have to walk.

DIEGO ESPINOZA: Diego Espinoza from Costa Rica. I saw the number of locked domains is around 1000, yeah? I know this is a low proportion compared with all your registries.

I have two questions. One is, in the time, how much percentage of domains you expect will be locked? And what is the main use of these locked domains? High-value domains? One company or person or individual [inaudible]? I don't know.

JAROMIR TALIR: For the second question, I don't have any statistics about that. I know that there are some high profile domains like Google or some banks. So I expect this. It's exactly domains like these, or maybe some [paranoid] normal domain owners that want to have this kind of security.

I didn't get the first question. How long the—

EBERHARD LISSE: The percentage of locked domains. What do you anticipate?

JAROMIR TALIR: We have about one million domains, so the 1000 is a really small percentage. But definitely this is not for anybody, and without this feature of immediate blocking and unblocking or unlocking, it's not easy to do this locking via this procedure to go into notary and submit some documents. So maybe it will grow right now.

EBERHARD LISSE: But if you want to reduce the number, just make it as expensive as we do and it's easy.

I actually wanted to start with the round table now because we're a little bit behind. Thank you very much, Jaromir, as usual.

I would like to then ask all the participants – [Mehmet], that means you, too.

UNIDENTIFIED MALE: How do you want us to – do you want me here, or do you want me…?

EBERHARD LISSE: You are the boss. I'm done.

UNIDENTIFIED MALE: Okay.

EBERHARD LISSE: Okay, we're having a small technical issue. That's not a problem. We're using two hand mic, which means that I would propose that Andrew, as the Chair, keeps one, and then whoever is speaking will keep the other.

Okay. Can all the round table participants take their seats?

ANDREW SULLIVAN: All right. Then we will. I want to thank all of the participants here. I am Andrew Sullivan, by the way, and I'm going to be your moderator today. Our plan for today is to have people say a few words about the implementation that they have with particular high points they want to discuss.

We've tried to arrange for most of the time to be discussion and so on, so I will remind once again – the poor participants heard me say this already 30 times on mailings amongst us – but I remind everyone that brevity is the soul of wit, so you don't want to go on too long. Then we'll get on to a number of conversations.

So in order, the order we're going to discuss is PowerDNS, then NSD, then Yadifa, then Bundy, and the Microsoft Windows DNS implementation, then Knot, and finally, BIND 9. That's the order we're going to have this discussion. I don't know if everybody is arranged in that order.

Then I have some other things, some various questions that we want to get to later on in the discussion. In particular – and I think this is something that maybe some of you can be thinking about as people are talking – I think that there are a few things we want to talk about.

To begin with, on the authoritative side, we know that we've focused a lot historically a lot on query performance, response rates and so on, but one of the things that maybe we haven't always focused on is what other kinds of criteria are important for performance? Maybe we need to think about that.

The recursive side of course is a different problem. There are problems with even the definition of performance for recursors, so that's something that I think it would be interesting to pursue today.

Provisioning remains something of a black art, and I imagine that there are operators here in the room who have experience with running more than one implementation and finding that provisioning works in completely and randomly different ways, depending on what the implementation is. So that's an interesting topic we might want to explore a little bit.

There's also this question of code diversity. We have a lot of history with code diversity and the idea is that we have many different implementations at the edge, but one of the questions that we ought to asking ourselves, of course, is how much of that code is actually shared? Are there underlying libraries that everybody has so they're in fact all subject to the same vulnerabilities anyway?

Conversely, are we in fact trading complexity for – operational complexity in particular – for something that isn't a real improvement, like for instance because everybody is talking to one another and everybody is referring to one another's implementations in order to see how things work? If everybody is actually relying on the same reference implementations, then maybe actually we're not getting the improvements we would like.

There's also a question about how different implementations deal with key management. So if DNSSEC is a hot issue for you, then that's an important thing.

Finally, to go back to the original performance question, there's also a question about the trade-off between performance and memory. If you're running very large zones or very large number of zones, memory footprint may be an important question, and that's one of the things that some people maybe need to think about.

So those are some additional topics that I have been contemplating and probing and prodding the various participants today to think about. In the meantime, to begin with, we'll get a quick overview from our various presenters.

So to begin with, PowerDNS.

UNIDENTIFIED MALE: Thank you very much. I will try to keep this brief. I will also try to keep this [inaudible] some kind of a vendor sales session. But very briefly, what this is is PowerDNS? PowerDNS is unable to go to the next slide – yeah, now we are.

People sometimes forget about this, so we're very happy that we've been invited here. We've been around since 1999. Our business model is to provide very good 100% open source software. So everything we do is open source. We couple that with the very highest-end 24/7 support for the people for who it really matters.

We have an authoritative server which powers around one-third of all domains. In some places, it's more. In other places, it's less. It can store data in databases, key value stores can base them off scripts, it can hand off queries over restful APIs. We can talk through smoke signals, if you want.

A few years ago we decided that DNSSEC was becoming sort of mandatory, but we realized that our user base did not to invest a lot of time in it, so we decided to make One Touch DNSSEC support, which has now sort of embarrassingly resulted in over 90% of all DNSSEC domains being ousted on PowerDNS.

Also, [inaudible] we have lots of scripting to, well, fix everything that's wrong. Then we also offer a PowerDNS recursor with the cache functionality, and our focus there is high performance, low latency robustness, which means that you should be able to forget that you have a PowerDNS recursor. Like authoritative, it comes with very powerful scripting abilities, which can sometimes be used to, for example, connect voiceover IP customers to a gateway that's nearby, instead of one that is at the other end of the country.

This is the graph of the DNSSEC adoption in various countries. The orange one is of course Dutch. This graph I think shows the value of cooperation between a software vendor and a registry. DNSSEC is scary to people to deploy, and rightfully so. So SIDN reached out to the registrar community in the Netherlands and says, "Look, we want to further the cause of DNS, and through SIDN's help, we were able to stay online with all the various registrars as they turned on DNSSEC," which is why the orange line went up rather smoothly, although on this scale you cannot see the various [1000] domain name dips. I didn't quite go right. These are the other countries as well. Not all of this graph is PowerDNS, but most is.

Here are three issues that I think are relevant for the discussion now and for later. I'm not claiming that we've solved them best, but I think they would be good subjects for the round table.

DNS provisioning. We can master name server and the slave name server and it will keep the contents of the zones synchronized. But what we do not have in DNS is a way to say, "This server will slave all our zones." So we cannot provision using some kind of standard protocol that this server should now slave that zone, or that this server, very importantly – I was reminded of that today – should now delete this zone. A lot of out-of-date content out there.

We've not managed to standardize this. There have been some efforts by vendors like Yadifa, who's done some admiral work. We've done some work. Commercial DNS providers have done work. But there's no way to interoperate on this front, and we would love to improve that.

So to further the situation, we've now also created an API, which is at least rather simple, by which you can tell PowerDNS over HTTP, "You should now do this to that zone." That has a built-in web server, so it's really easy to integrate. As long as you can talk HTTP, you can tell PowerDNS what to do.

What this API also does is you can add zones, but you can also add records to zones, and if you try to add a five-digit IPv4 address, it will reject that. If you try to do various impossible DNS things, it will reject those changes, which is nice because the people who make the web-based APIs do not want to implement all the DNS-checking rules because it's very hard to do from PHP, which is why we rejected from PowerDNS.

We think this is a second important part because the amount of people that enjoy editing zone files with a text editor is decreasing rapidly. But the next step is the amount of people who want to edit a name server configuration file is also decreasing rapidly. As knowledge is leaking out of Internet service providers and moving their way to cloud providers, it is important that DNS software becomes easier to use and we stop glorifying VI as user interface tool.

So we built a PowerDNS web-based API web-based GUI on our API. It's rather simple. Well, you can manage a whole bunch of DNS servers from one interface. You have graphs that are individual and aggregate, and statistics, create, remove, the dead zones – the whole shebang. We've been making this over the past year because we are mainly a software company, but every once in a while, we engage with a customer directly to just figure out if our stuff is still good enough because every software vendor has the problem that they don't run their own software.

So over the past year, we've been developing this PowerDNS control user interface. The name might still change. Looks like this. It looks like you would expect it to look. This is your list of servers. You can edit them. There's little graphs. You can do flush cache, or if you have a domain name that needs to be gone from the cache because you just changed it, it can go in one place from here. You can [sit] down and do all the things that you expect to do.

Here are some graphs where you can actually see a denial of service attack in action. Normal graphs don't do this kind of thing. This is an attack from a 400,000 queries per second inbound to us, which means 200,000 queries per second outbound. You can see it [inaudible].

So this is the first trend that we're highlighting: provisioning. The second one is DNS software needs to be easier to use.

This is the third one. Denial of service attacks. DNS is being used more and more and all our customers are under constant attack. These are not necessarily the million queries per second attacks that everyone is talking about. These are maybe 4000 queries per second, but naively translated, a 4000 queries to a resolver can turn into 40,000 or 100,000 per second to an authoritative server.

No one has time for this, and on one wants to. Things are moving and I don't want them to move. Is someone controlling this computer? Okay. Yeah. Okay.

There are some conflicts in the handling of a denial of service attack. If you have an incoming attack of a million queries per second, it's tempting to answer those all million queries because it's nice because if you answer all incoming traffic, you answer all legitimate questions as well, so your end users might not see any disruption of service.

Then we go to the second in bold: do not overload third-party authoritative servers with queries. In your laudable attempt to answer all queries, you are flooding the net with gigabits per second of DNS traffic that they didn't ask for.

The third one is the same thing. You might be part of a reflection attack. So if you build the server that does ten million queries per second, you've just enabled a ten million queries per second attack, and thank you for that.

The fourth thing that people want from the server is not that it doesn't call them at night. They want to have a zero maintenance server. DNS is typically be near zero maintenance, and even though we now have denial of service attacks, people still want their name server to be zero maintenance. It's basically impossible to square this circle. You cannot have it always.

The other problem with that is – I have some pictures of cute animals – when you ask people to upgrade their name server infrastructure, they're typically not very enthusiastic. It's very difficult to get people to install new name server software, which represents [the nice turtle].

Meanwhile, as we heard in the previous presentation, attackers have [decision skills] of five minutes. They can deploy a whole new infrastructure in five minutes. So the decision loop of an attacker is way faster than that of the defender. Also, the defender goes home at night.

We see that as a problem, and I'll be really quick now. We've been asked to improve PowerDNS support for denial of service resistance measures. We did not implement the response rates limiting stuff up until now, and we were kindly asked and well-motivated by our friends at Dutch ISP at ISPConnect and the Dutch SIDN registry if we could pretty please implement [RL].

Because we were late to the party, that allows us to implement that goes slightly beyond response rate limiting for denial of service prevention, and in consultation with users, we found out that we need to break through this cycle where you need to deploy a whole new version of your name server software to fend off an attack.

So we've made the whole logic response rate limiting dynamic. It's in [inaudible]. It's completely flexible. In the interest of time, I'm not going to tell you more about it, but the goal is offer responsible response rate limiting functionality out of the box while retaining the ability to change strategy if you have an attacker that changes their strategy in five-minute timescales. Thank you.

ANDREW SULLIVAN: Thank you so much. Rather than take questions right now, unless they're clarifying questions, I'd like to hold them to the end. I don't see anybody putting their hand up, anyway. So we should move to NSD next.

UNIDENTIFIED MALE: Okay. [inaudible] I had some slides as well, but I don't think they came up in the thing.

ANDREW SULLIVAN: They're not on here?

UNIDENTIFIED MALE: So I don't know how this works. Maybe for time reasons I should try to do it without slides? If you're looking or not, I'm just going to fire away.

ANDREW SULLIVAN: I don't see how they're here. I don't see where they are.

UNIDENTIIFED MALE: No worries. NSDs from [inaudible] Labs who does thing that if you're building name servers you should build something that you're good at, and then do that perfectly.

So then we come up with the NSD, which is an authoritative-only name server to be lean, clean, simple, very fast. So we have these building blocks in DNS, and we have the authoritative name server. We have a recursive name server. We're making different software for each functionality.

So NSDs or authoritative name server, as I said, is open source, of course. It's created for biodiversity in the open source name server world at that time, and there weren't that many open source name servers.

Yeah. And it's focused on query performance and something we might want to discuss later on because we're focused on a lot on the performance which comes at a cost, of course, and usually that comes at a cost of memory. So what is actually more important for you?

We have a new version, which is NSD 4. The main difference there is that it can deal with lots of lots of zones. In NSD 3, there's a bottleneck for that. 4 fixes that.

What does that mean? You don't to restart your name server anymore if you want to add or delete zones. We have introduced things like patterns, so if you're very much a similar zone, you can just create a pattern and then the configuration is much easier.

It's current version is 403, so I advise you take a look at it if that's something you're interested in.

I said earlier we're believing that making the building blocks, so I just also want to focus on Open DNSSEC, which is also the registry part if you're doing DNSSEC.

Typical use case for NSD would be one of your public interface secondary service, so to say**.** Open DNSSEC would be in your infrastructure as a bump in the wire between the provisioning system and the secondaries. It could also be part of your provisioning system closer to the backend, so to say.

Open DNSSEC you can get your zones signed. It's very easy once you have set it up. Of course, you don't have to take a look at DNSSEC maintenance anymore, so it gives you lower maintenance than otherwise.

You can do zone signing on transfers, on files. Just listing some features here. It does key management through an [HSM], so through the PKCS11 interface, so you require an [HSM] or a software version of [HSM] which you'll also provide.

That's my quick notes I have here, and since I don't have slides, I'll save the rest for the discussion.

ANDREW SULLIVAN: Thank you. I have to apologize. It must by my fault that the slides didn't get included. So you can blame me.

Next we have Yadifa. Has this come up as well?

UNIDENTIFIED MALE: [inaudible] found it just now.

ANDREW SULLIVAN: It's actually under my name for some reason.

UNIDENTIFIED MALE: It's under your name, okay. That sort of sounds nice.

ANDREW SULLIVAN: It's just the file name, though. I don't take credit.

UNIDENTIFIED MALE: Here you go.

UNIDENTIFIED MALE: Okay, thank you. More of the same, I would say. Thank you. More of the same. Yet another domain name server. So let's go to what is Yadifa? Yadifa is a name server implementation. It's open source. BSD license.

Basically, you get the parts. You do with it whatever you like. You make something new of it and in whatever way that you do it, it's fine by us.

It's written from scratch, so all mistakes I will own, or so we hope. That comes back in the next slide what I really mean with that, and is developed by EURid, which is the .eu registry. And I, as Andrew, do not take any credit for writing one single line of code of this thing. If you want to meet the guys, they're sitting on the front row there, so you can actually talk to them later on.

What did we try to do with Yadifa? I just put down a few slides with some design goals and different priorities. Our first priority was that Yadifa would actually have to be a public slave, and it would have to be a top-end replacement for any of the others sitting actually on the table be it BIND, NSD, PowerDNS, or anything that you can imagine that actually does DNS queries.

It should be obviously RFC compliant. If you go a bit more into technical details, it should be RFC compliant. It should be authoritative. It should support DNSSEC, at least from an authoritative point of view. It should do [inaudible] as a slave. As it is a public slave, it should support zone transfers as a slave.

Performance is important. Some of you might have seen the slide wars between different vendors some time ago with the number of queries that you could get from a single CPU code or multiple CPU codes and things like that.

Suffice to say, I think you're fast enough. The point is that most of these name servers implementations actually support answering queries, 100,000 queries per second. So the point there is that it should be performed in a sense of it should be speedy enough and it should be low memory enough and all things related to that.

Another requirement, it [should just have zone] files as a storage packet. [inaudible] differently. Interoperability is important. The whole point is implementation diversity is important. Our goal here is you have a set of name servers running out there as public slaves. BIND, NSD again, any of the others, I should be able and you should actually take a healthy mix of name server implementations.

If you come back to the previous slide where I said all mistakes I will own, the whole idea is that not the same vulnerabilities will appear in all these name server implementations, if there are any vulnerabilities, obviously. You define reality here. Well, I am.

Anyway, the second set of design goals is one step further. If you have your public slave, then maybe you want to change your hidden master also to its Yadifa, and that's then of course the second set of priorities or design goals. There is should support dynamic updates. We have heard about this in previous presentations of CentralNic where they actually are doing the same thing now. So Yadifa supports that. You can send it dynamic updates and it will actually incorporate that in its running zone files.

Also it should be doing continuous online DNSSEC singing or whatever you want to call that. In any case, you send a dynamic update and it will take care of generating the signatures or updating the signatures as it deems necessarily. Obviously, it needs access to the private keys for that and also and so on.

Third set: operational flexibility. Obviously, once we have a set of name servers running. Also Bert talked about it. You don't want to look at it. It just needs to keep on running. But if you want to add some zones, remove some zones, or do something, you don't want to bring down your whole infrastructure to do so, but you want to do that on the fly. You should be able to add a zone, remove a zone without stopping answering queries for the existing zones.

That's what we are currently implementing in Yadifa, the next release that's coming, hopefully in the coming week or two. We have it already in beta running in our labs. You can add the zone files and then you can just tell it, "[inaudible] conflict file" so you can edit the conflict files, and then you can tell it, "Reload the conflict files first and do what is necessary to bring online the new or removed zones," or whatever it you made changes to the conflict files. That's one step.

The other step, also alluded to by Bert is the dynamic zone provisioning. We have been twiddling around with actually allowing a default to actually configure itself via a network protocol, and we have chosen to use a DNS protocol and some extensions on the DNS protocols in terms of other classes and things like that, so we'd actually be able to, like you, configure the zone itself, add and remove resource records actually over DNS, add or remove zones and configure the zones itself.

The whole idea would be then in the end that this would be a IETF standard, and that obviously all name server vendors would implement the same thing so you get the same sort of interoperability in terms of dynamic zone provisioning in different name servers.

Fourth set: tools and libraries. The whole idea is that, like you have now tools like [dig] and DNS Lookup and all sorts of things that you would have another set of tools, some control utilities to actually control running demons, some tools to be actually be able to send or generate dynamic updates, obviously to be able to do some DNS queries.

Basically, you will have a set of building blocks, libraries, on which you can build other tools. I just made one slight thing here on what we are actually using in the – I don't know who DGH was, but he was raising his hand for one reason or the other.

What we have done for the .edu registration system is basically built on the [inaudible] to have an application that listens to our core registration database and message [inaudible] basically extracts from that new domain names or changed domains names and actually feeds that as a dynamic update to the hidden master.

The whole idea is that these libraries will be sufficiently to a lot of people and easy to use to actually build whatever constellation of DNS-based software that you might have in mind.

Last but not least, the other side – the recursive or validating side – we haven't really tackled that. It might be interesting. We're not doing that yet. That's not to say that we won't do it. It's not just something that we actually had a lot of time to work on. For the moment, we're actually concentrating on the authoritative side.

Last slide. The end is near. The key message I think is if you have a DNS infrastructure out there, you should have, in our opinion, different DNS implementations, and Yadifa is just one of the choices that you could make to run part of your DNS infrastructure. Thank you.

ANDREW SULLIVAN: Thank you. I don't see any clarifying questions. No. All right. So next we have Bundy.

SHANE KERR: Okay. Hi. Thanks, Andrew. My name is Shane Kerr. I was involved with a project called BIND 10 for a number of years, which was a project to implement a new version of [ISC's] name server, BIND 9. [ISC] decided to cancel that project at the beginning of this year, but they were nice enough to help myself and a group of other interested people in moving the code to a more community-based effort, which is the Bundy Project. If you to GitHub and search for Bundy DNS, you'll find the code repositories up there. We've also got a website. If you Google it (bundy dns), you'll find it.

We started as a code base with all the code that we inherited from [ISC]. The goals were a very flexible and forward-looking DNS server, which was highly configurable. It's interesting looking at all these things about adding provisioning stuff. We had that from the beginning [inaudible] XML-based provisioning that you could use over a network and stuff like that.

What we didn't have is a small, simple, usable DNS server, so if you're interested, you can go – I did a presentation at the RIPE meeting about the many reasons why BIND 10 was ultimately not successful, but we're hoping that Bundy is going to be useful in ways that BIND 10 wasn't.

Now, because we're a community effort right now, we don't have a single company or a group of companies directing the effort, so each of the individual contributors has their own goals for the project.

One of the contributors, for example, is very interested in the user interface aspect of it. We have other people who are interested in using the core technology to write a very small, fast DNS server that fits their needs exactly. Other people are adding bits that are missing from this server as it goes.

We don't have a clear vision right now. We're hoping that one will kind of coalesce over the next months or years. We don't have any specific drive to do that.

My own thinking is that, if we're successful, it will be useful for people that want to do things that are unique in the DNS world and want a basis to start with, instead of taking it off-the-shelf solution. If you know that you don't fit in this space with an off-the-shelf solution, I think Bundy is going to be a good starting point for you. I have really solid DNS library and a very straightforward, within the requirements that we have, infrastructure for how we fit all the pieces together.

So anyway, that's about it. I don't have any slides. Come and check it out if you're interested, and you can come and talk to me. I'll be here today and tomorrow.

ANDREW SULLIVAN: Thank you. So next up is Microsoft, but I'm not finding the slides here. Are they on someone else's computer – oh, you have them? You can put them up. Oh, yup. I think they're there, right?

UNIDENTIFIED MALE: Right.

UNIDENTIFIED MALE: Because the remote participants can't here you unless you speak through the microphone.

DAN YORK: Just a quick question on while we're waiting on that for Shane. Shane, not knowing, how much of Bundy – how much similarity does it share with BIND and BIND 9.

SHANE KERR: Almost none. Yeah, well the original intention was to reuse as much of the code base as possible. It turned out there wasn't much that was possible. One of the goals of BIND 10 was to have loose coupling between the components, which is fancy computer science way to say each one should be as independent as possible.

In BIND 9, that seemed to be an anti-goal, so it was very difficult for us to pull out individual components. So in the end, we ended up basically rewriting everything. So, yeah.

MEHMET AKCIN: Hi, everyone. My name is Mehmet Akcin. I've met with many of you all at ICANN for many years.

I'm going to cut it fairly short and I will skip a lot of slides. I joined Microsoft about nine months ago, and – sorry, excuse my voice. I yelled to much when the US scored last night.

I joined Microsoft about nine months ago, and when I joined I was like, "Okay, so these are your responsibilities. These are the name server infrastructure that you need to run in order to support one plus million physical servers that are deployed across the world for Microsoft operations."

When I looked at the infrastructure that was there, I was like, "Oh. This is all Windows DNS server. I have never run Windows DNS sever in my entire life, or maybe last ten years of my career. But I guess I can just pick half of these servers, install BIND or NSD or something like that, and I'll be fine with that."

So I started looking more into it, and I started thinking that it would be an easy way to solve a problem. Then I started looking more in depth into it. Why did I think that would be an easy solution? Because I really didn't know what capabilities that were behind Windows DNS server. There has been previous experiences where Windows DNS speakers have been in these kinds of conferences. They speak about the performance and the capabilities and all that, but can you do better? Perhaps.

So this is one idea I wanted to start with. I'll go to Windows DNS server now. Basically I've come to the conclusion that – so I have this software that I own, and I can pretty much go talk to the developer team and every other team that makes the operating system work faster and see what we can fix.

This is a basic result of it. I will skip these real quick. I don't know how we can actually come to a conclusion of the usage numbers, like the ISO attorney present, but other people might claim 50%, 20%. I don't necessarily know how can we come up with the conclusion that this is the amount. All I want to say is Windows DNS servers are heavily used in active directory deployed environments, small, medium, large businesses, authoritative layer as well for several large corporations.

The goal here is that we actually want to deliver a DNS sever that can work with everyone else. We want to make sure all the RFC compliances are closely monitored, adopted. If you want to change something, the way it does work is we go back to IETF and discuss and make sure that everybody's aware of these kinds of changes, so people can also adopt those changes.

The point that I want to make that I think is very important for Windows DNS is the ease of use. It is a graphical user interface which kind of makes life easy for someone who doesn't necessarily want to spend too much time on DNS. And it does automate a lot of things. But for those who really don't like graphical user interfaces, just like myself, I like PowerShell. It is fairly similar to terminals that we have been using for many years.

Then there are several functions that we have been in the community for the last six months very actively engaging, very actively asking people, "What can we do better? How can we fix things if you think it's broken?" and basically we were given suggestions based around NSID rate limitings [inaudible].

I can neither confirm nor deny they will be coming up soon, but I can say that we have received these suggestions, so if you have any other suggestions for Windows DNS to support or adopt, please feel free to talk to me right now, or we have a booth here. You can come find us as well.

More features. Dynamic update, integrated with [inaudible] and [inaudible] and DNSSEC is co-exist. Some DNSSEC information, DNSSEC has been part of Windows DNS server since 2008 R2 and we have actually sit down and rethink about it much more closer to operate with other operators in Windows 2012, especially R2. So we have NSEC 3 support, and we work with so many different vendors of hardware security modules, providers – HSM providers.

Again, supporting online signing, active directory integration, dynamic updates, and improved performance. So will quickly skip those. And basically everything you can do in GUI you can do without GUI as well. So if you are a GUI hater just like me, you don't really have to use the GUI.

Just some key points real quick are the resigning on static and dynamic updates, automated key rollover, signature refresh, automated updating of security delegations, and distribution of updating transanchors.

Real quick about performance. So we have this team that's super-enthusiastic about speed, and when I discuss with them actually, they lecture me so well about that speed doesn't really matter too much. Speed is not the only thing that matters. There are other things that matter, like the documentation, having that documentation. Be reachable and available for a big amount of people, like just one idea, I think those Windows DNS server documentation is today available in twelve different languages, every piece of set up – step-by-step, DNSSEC guide, everything – and we are working on expanding that number to 20. So this is something else that actually helps Windows DNS I think to be a diverse alternative to every other people that is actually operating a DNS server.

Performance-wise, just real quick numbers. This is a test that we did, and I'm more than happy to talk about the details, and I had those details. I intentionally removed them just to save some time, but they're about one million zones with 10-25 different resource records reloaded on a Windows 2012 R2 server, running on 64 gigabytes of memory. We have done some internal tests and reached these numbers. We are working closely to do better comparison tests to find ourselves and compare ourselves with our colleagues in the industry that we are looking forward to [inaudible] close with for one Internet.

DNSSEC performance here. I have some numbers, as well. Again, we can talk about these, our singing performance with NSEC, NSEC 3, and different scenarios. Again, I'm more than happy to discuss these.

So summary: easy to deploy, smart default. Smart default is something that we really come up with the Windows 2012 R2. Again, this team in India just wants to make everything easy for someone. Here in Turkey – although we are not in Turkey – here in London, this small office running a small shop, but once they use these functions and that doesn't really necessarily have interest of time to go ahead and find out what the defaults are, so we made sure the defaults are within the suggested rates of IETF standards.

Automated management for day-to-day operations. One of the big concerns was that DNSSEC would actually involve some – you will need to do some additional things, and there are certain things like changing your KSK and updating that to your register is still part of the thing that you need to do, but we try to automate as many things as possible.

RFC compliance, high performance augmentation is available in twelve languages – I should have updated that – and we recently opened a mailing list, and the mailing list was super-active recently in the last couple of days.

So many great questions, so many great suggestions just flowing to e-mailing list about DNSSEC, coming with suggestions. We have full-time people basically just replying to that e-mail. They say, "I'm very grateful." It's not really their job but they're so enthusiastic to be able to come up to this forum and actually present something – a diverse alternative to every other operating system and service.

So any questions? You can find me here. We are also going to be at the DNS-OARC event in Los Angeles. Thanks for this opportunity.

ANDREW SULLIVAN: Thank you. All right, if there are no clarifying questions, then we'll ask Knot next, please.

ONDREJ FILIP: Okay. Good afternoon. My name is Ondrej Filip. I'm from CZ.NIC, which is domain name operator I want to say for domain .cz, but more importantly we have a very huge R&D department which develops a lot of software, especially open source software, and right now one of our flagships is called Knot DNS, an authoritative DNS server.

UNIDENTIFIED MALE: There's a button down here.

ONDREJ FILIP: Oh, there's a button? Thanks. Excellent.

So Knot DNS was introduced in ICANN in Prague. It was ICANN 44 I guess. Those who attended meetings regularly know about it, but for those who doesn't, here is just a one-page introduction.

So it was developed for us, for .cz, as a high performance [inaudible] authoritative DNS server. As I said, it's free open source and it's written from scratch. It's supported actively by our registry and it's run on our own DNS servers, so we believe it's stable and we operate it our own selves and by some other registries.

Of course, as everybody mentioned, we follow all the standards and we try to fix bugs as quickly as possible. It was designed for nonstop operations, so it does the runtime configuration and stuff like that.

It's usable in all possible scenarios like root server, top-level domain, or DNS hosting. So a company that has millions of zones.

There are some new features, which I will discuss later one, like DNSSEC automating and signing and some pluggable modules that really enrich this software.

A little bit about history, and more importantly about the road map. The current stable version is called 146. Actually, it's 147 now. Anyway, so it's full-featured authoritative DNS server. We are not making this software as a Swiss knife. We'd rather follow the UNIX way: make a single task and make it good. So we don't have like GUI – nothing like that. It's a pretty simple configuration based in text and commands and stuff like that.

What we added recently is DNSSEC automated singing, which again I will discuss later. The new version which is currently in the release candidate phase – so it's going to be released soon – is called 1.5. It doesn't have many changes, but one important thing, it's completely refactored, which is quite an interesting thing, and I will show you what I means, actually.

We are working on version 1.6, where we would like to do some real-time DNSSEC signing and also some key management utilities.

Now back to those features that I touched on a little bit. DNSSEC automatic signing was meant as a technology preview, but it's very stable and some people started to use it, so you can say it's a stable feature now. It has a very simple configuration. You just configure a zone, define where's the zone file, where are the keys, and you just put the closed DNSSEC enable on, and that's all. It does all the magic. It signs the zone and resigns it so it's up to date. It's very simple for anybody who doesn't want to really play with DNSSEC. Easy feature.

Another new thing which is 1.5 that's in the release candidate – as I said, it's going to be very soon – we implemented pluggable modules. There are hooks in query processing systems, so you can hook into the system and do something with that. There are different possibilities of how you can play with that.

You can use it for example for split horizons so the DNS server answers differently when the query comes from a different country or different IP. You can use it for some load balancing and stuff like that.

But just the possibilities we implemented actually two features. First is called reverse and forward resource record synthesis. I have a slide about it. The second is called dnstap. That's something that Farsight came with – the way how to do a passive DNS, how to store data from the DNS flows. So it's just a simple way how to support this initiative of passive DNS.

Now back to the synthesized resource record. Something we were approached by some ISPs that the develop IPv6 and they faced a strange problem. The IPv6 address space is vast, so it's not possible to configure reverse records manually. The zone file on memory consumption would be enormous. But on the other hand, customers wants to send e-mails from their DSL lines or end user homes, and some of the MTAs are checking for reverse records, so they are rejecting e-mails sent from IP addresses that do not have reverse records. So that's a problem because, as I said, the space is huge and there are plenty of possibilities.

So those customers started to complain that it's bad, that IPv6 actually made our situation worse. That's why we came with a feature that synthesizes those query responses. I will show you that in a simple configuration. You don't put a zone into the configuration, but you just put a close [sentry] code and you do, for word generation, for some IPv6 space or you can of course you it for IPv4 as well, but of course the usability there is much smaller. Then you do the same close for the reverse records.

Knot DNS then does all the magic. If it receives a query from some IPv6 address, it responds accordingly, and customers are happy. The only limitation is we currently do not support DNSSEC in this thing, so that's something that we're working. In 1.6, we expect that those queries will be signed as well.

Here's some example of output. We have [inaudible] command. As you can see, I chose some random IPv6 number and I got the answer and the same in the opposite direction. So, very, very handy, especially for ISPs that are largely deploying IPv6.

A little bit about DNSSEC support plans. So some things that's going to be in the future. We would like to separate the DNSSEC stuff to separate the library which is temporarily called LibDNSSEC. I'm not sure if we're stuck to this name. We want to switch from OpenSSO to GnuTLS. We decided before heartbleed attack, so there's no more relation there. But now it's the view that we should do it.

We plan to support hardware security modules using standard PKCS interface and we would like to add some new key and signing policy tools, again, to improve the DNSSEC support.

Again, we would like to the online signing, which has the main sense with the synthesis records, because then it makes sense to assign online. But of course, it can be used for very large zones that are created just a few times.

I also mentioned refactoring. This is the result. As you can see, the number of line codes is actually dropping, so we are really removing any stuff that isn't necessary, trying to keep the source code very simple, very readable, and we rethought every piece of the code, just to be as optimal as possible. So unlike many projects that are growing in time, we are really decreasing the number of lines, or at least we are on a stable level while getting some new features.

Last thing I would like to mention is the benchmarking we do. Just to understand what's our position now and how we perform, we did our own testing. We decided to test some common DNS servers like BIND, two versions of NSD, PowerDNS and two versions of Knot, [like stable one and the release candidate usually].

The benchmark is open source, again, of course. Here's a URL of it. You can download it and try to use it. We use just the [inaudible].

What is this about? We have three servers. One is sending packets. It has some real data that was captured. It sends to a server and server replies back, so the data is not synthetic. It's real, so there are some queries that can be answered or they can't be answered, so it's a really good mixture of what happens in real time. The list [inaudible] the percentage of response that came back. Then we compute how successful the DNS server is under which load.

We created several scenarios, like root zone set up, two TLDs – TLD with and without DNSSEC – and DNS hosting, like 100,000 zones and one million zones with two different hosting companies.

Here's a URL for results. But just to show you that this looks – this is, for example, the graph of root server scenario for response rate. You don't have to trust our numbers, so you may ignore the implementations. You can just see the progress being made between our two versions. So it's in TLD scenarios there. There are others, so go to the URL if you're interested and you can play with this.

Also, we are trying to measure start-up time and memory footprint, and I intentionally took one scenario we are not the best, but again shows the progress of Knot DNS from 1.4 to 1.5, so you can see the reduced memory consumption really rapidly. It's much bigger progress than in response rate because there we are very close to what we believe are limits of those machines, actually. So that's about Knot DNS. Thank you very much.

ANDREW SULLIVAN: Thank you. So I see no clarifying questions, so I will ask finally BIND 9 to come and present.

BRIAN REID: Can you do my slides from there or do I need to walk down?

ANDREW SULLIVAN: I probably can.

BRIAN REID: Then could you, please? Is this microphone loud enough? Okay. I'm Brian Reid. I'm at ISC. I guess I've been there about ten years. I'll talk while he's getting the slides set up.

ISC is a smaller-than-you-think non-profit that's been teetering on the edge of bankruptcy for as long as anybody can remember. Every now and then I run into someone at a conference who thinks we're rich, but who knows where they thought that?

Why don't you move to the first real slide? Yes, BIND use cases. Is there a way to align it to a page boundary?

ANDREW SULLIVAN: Oh, well, hang on. I'm doing it wrong, like usual. There we go.

BRIAN REID: Better than I. BIND is the Swiss army knife of DNS software. It might not be the best way to pry open a paint can because that's the screwdriver blade, but it works.

BIND will work in any use case. It might not necessarily be the best solution for every use case, but you can count on it. Some of the recent BIND features that I might talk about only work with some use cases. Some are only good in recursive situations. Some are only good in authoritative situations, but it's all in there.

We have absolutely no idea how many people use BIND because we have never put in a phone-home code, and we've it away and it's packaged with operating systems and whatever.

We do know as a factoid that 35,000 copies of BIND have been downloaded in 2014 from our sever, but it's mirrored in 30 places.

Over at the right is something that's circular on my screen and probably oval on yours. It's a pie chart, also an I-chart, I guess, showing the kinds of organizations that pay us money to provide support. That's kind of our business model. We give away the software and then sell support. It works, but I'm going to have to turn to my laptop to turn the numbers because the ones on the screen I'm only ten feet away and I can't read them. I can't read them on the laptop either, but there's Enterprise and there's TLDs and there's OEMs –

UNIDENTIFIED MALE: Telco.

BRIAN REID: Oh, Telco. Yeah. There's stuff, and they're all different and they all have different use cases, but what they have common is that organizations who fit into one of those pieces of pie have all signed contracts with us to provide 24 by 7 support. There are lots of reasons why they do that. Next slide, please.

One of the things that BIND has done for as long as I can remember is implemented almost everything almost first. If it's a feature that's in an RFC and it's made it to far enough along the track, we implement it both because we think it ought to be there and because BIND is a good place to test features to see if they really ought to be there.

One of the things that that means is we aren't necessarily the snazziest at everything because as several of the panelists has pointed out, if you go second, you can do things that the person who went first didn't think of just yet. So when we redo something, we'll copy ideas from you guys.

But it struck me that editing configuration files is actually a really good thing. It seems boring and old-fashioned, but it's auditable. It's archivable. If you have a config file and that config file was used to initialize your server, then you know what your server configuration is. But if you use our various features for dynamic, online real-time configuration, which are fun, they don't necessarily leave you with an archivable object that specifies your configuration. That could certainly be done but we haven't done it.

We're a great believer in diversity. We try to make sure that we work with everybody else's name server, and so we support all of the configuration update mechanisms that seem to work with the other systems' DDNS. Next slide, please.

These are a few recent BIND – well, [views] aren't recent. They've been around forever. Inline DNSSEC signing is in either the newest or the next-to-newest release. RPZ you've probably heard of. RRL has been out there for maybe a year. Dynamically Loaded Zones (DLZ), that started out as a user-contributed feature and it's now part of real BIND. One thing that is not – I don't think we've shipped this yet – the resolver prefetch of expiring data. But those are just a few of things that either just in there or are good to have. Next slide, please.

I didn't make this slide and I can hardly read it myself so I might just gloss over it. We have lots of DNSSEC support. We're always working on more. The two things that we're going to release in the next edition are key management scheduled rollover, and also negative trust anchors. We have negative trust anchors working in a private branch, but not ready for primetime. With luck, nobody will need them anymore in two years.

The fact that we can import keys from other places and use HSMs means we can integrate fairly neatly into many but not all key management structures. Next slide, please.

BIND is a general purpose tool. It's universal, and if you want to use just one tool, BIND is the one to use. As I said, we work hard to ensure that BIND correctly implements every new RFC. So there's a lot of RFCs, so BIND has a lot of features. We occasionally have people make fun of us for the number of different features in BIND, but they're in the RFCs, so we think we ought to do them.

If you're running a large-scale, mission-critical service, then we believe the software heterogeneity is ideal. You should use BIND plus one of the other fine systems that you've heard about here.

We offer support for BIND. If you have an enterprise for which DNS is mission-critical, talk to us about paid support for any of these other packages. We'd have to talk to the package implementers, but we can probably set something up because support is different from software. Support keeps your upper management happy and your insurance company happy and lets your SysAdmins have a little bit more sleep sometimes.

But you really should be using more than one different kind of DNS software. It makes sense for BIND to be one of them, and lots of choices for the other. Thank you.

ANDREW SULLIVAN: Thank you very much. We have 31 minutes I guess of remaining time in which we can have some discussion here. So I would like to open the floor if people have issues they're passionate about and want to talk about. We'll start that way.

JAY DALEY: Hi. Thank you for the wonderful presentations. That was great. A number of you mentioned provisioning protocols. What's it going to take for those of you on the table all to sit down together to invent a great provisioning protocol and release it all in your implementations without us having to go through the pain of the IETF and people saying, "No, you're not allowed to do that"?

UNIDENTIFIED MALE: Ask the man sitting to your right.

ANDREW SULLIVAN: Though I am not an implementer of this, some of you know that I work for Dyn and we run some DNS zones. I would like to echo that there were several of the presentations where people talked about provisioning being an important feature, but actually provisioning zones in heterogeneous systems, as far as I can tell, remains a nightmare, and you always build new additional stuff on the side that does it. So it would be great if we had one way to do this. I know that Shane suggested that he thought a lot about this in Bundy, so maybe he wants to start.

SHANE KERR: Sure. For full disclosure, I now work for Dyn. Certainly, one of the issues that we had with BIND 9, which we decided to tackle head-on in BIND 10 was making sure that everything could be changed at any time, so no restarts and things like that.

With that as a requirement, this kind of Runtime configuration kind of fell out automatically, so we do everything in Bundy with simple commands that any module can support, so adding zones and things like that – all the basic stuff.

There was no standard way to do this when we did. There's still no standard way to do this. It seems like we've got at least three or four implementations here today if you pull [inaudible] realize that the provisioning sucks and have implemented their own improved ways to do it.

I think on the Bundy side, all that it would take for us to implement a standard is for there to be a standard. I'm not sure how to get there. I think none of the vendors have a lot of financial interest in pursuing that, because why would we make it easy for people to use other people's software? Speaking as a former vendor.

And I have to say I think the IETF would in principle be the ideal place for this, but I think in practice, the DNS culture in particular with the IETF is really broken right now. It's very easy to say no and stamp your feet and hold your breath and prevent things from going forward and very difficult to make even very simple protocol changes. So that's my take on it.

UNIDENTIFIED MALE: [inaudible] and then Brian.

UNIDENTIFIED MALE: I think it's a great question. Thanks for raising it up. One of the great ways to push a suggestion in organizations, the organization I work at least I can speak of, is to demand it. What you are raising and many customers echoing the same thing is definitely heard.

I think one idea is we need to identify where we can develop these operational practices. I call it documentation standards. I keep hearing that IETF might not be the right place. So is DNS-OARC the right place? Where is the right place? I think if—

JAY DALEY: [inaudible]

UNIDENITIFIED MALE: Right. So –

ANDREW SULLIVAN: For the record, Jay said this table is the right place. I think he meant that the implementers, not me.

UNIDENTIIFED MALE: I have some great news. We are meeting later today at the bar, so maybe we can work something out.

UNIDENTIFIED MALE: Jay's paying.

BRIAN REID: We have a lot of big enterprise customers, people who have 100 employees who are running their DNS system, and what we've discovered is that a provisioning system must be part of the corporate work flow, and they don't want us to put in a provisioning system because they already have one that they've built their way, so their need is for our product vying to be malleable enough that they can use it in their workflow in their provisioning system. We've been really startled by how many different enterprises actually do it that way and don't want us to give them yet one more obstacle to their workflow succeeding.

UNIDENITIFED MALE: Yeah, I think actually standardizing provisioning is a good idea. It's not so necessarily promoting the other software. It's promoting biodiversity. It can use the same provisioning for all type of software name servers and it makes it easier to switch from one to another.

I do think that IETF is the correct place to do it, although I think, if I recall correctly, there were two earlier attempts to make such a name server control protocol that died.

What the reason for that is I can't recall really. I guess the bottom line is we can wait to come to a form of consensus. I don't think I won't say people on this table have presented suggestions how to do provisioning, and although there can be a good attempt to come to a standardized way of provisioning, you have to convince the other implementers that this is the correct way to do it, and I think, well, there were a lot of discussions, and again, then finally no consensus, and the proposal died again. So I see some sort of, well, repetition going on.

UNIDENTIIFED MALE: Well, let me then state one thing. The IETF in the world of dreams lived on working gold and rough consensus, and it was wonderful. We've seen several attempts to converge on an idea, yet we have not involved working code in that.

Maybe we, as PowerDNS, we've been extremely surprised by the rapid update our HTTP-based API. Any initiatives that comes from DNS people is likely to involve the DNS protocol because on the tremendous amount of ways, it makes enormous sense to use the DNS protocol to update DNS.

For example, in an ASCII-based protocol or a UTFA-based protocol like HTTP tends to be, you can insert the IP address 1.2.3.4.5, which you cannot squeeze into a DNS-based update. However what we have seen with our dirty use of HTTP, we have seen tremendously rapid uptake of our API. People just take to it like fish to water because we did the [inaudible] the JASON thing, and it takes off like wildfire.

So while we would like to standardize, I'm instead predicting we will do something else. I'm afraid that we will go on with our own provisioning API since it has a little momentum behind it, but simultaneously if anyone says, "Look, you're doing it all wrong," we will be promised to take that into account and see if we can sort of prevent freezing-in mistakes we make now.

But we're not going to wait for any further standardized action, but anyone is free to join in and to talk to us and use our API, and we're willing to adjust our API, even for non-PowerDNS needs.

This is mostly – and I'll finish on that – based on the surprisingly rapid uptake on the things we wrote, which gives me some confidence this is the right way to go.

EBERHARD LISSE: I'm wondering what's preventing you seven there to get together and agree on something.

UNIDENTIFIED MALE: We're not the users.

EBERHARD LISSE: Sorry, but it's not the IETF that is preventing you from doing it. But I'm just wondering if you seven sitting there would agree on something and implement something and it would be out there, maybe some users would use it.

UNIDENITIFIED MALE: I think the things that we do not agree on is the quick and dirty –what [inaudible] was just talking about. The HTTP interface which is basically a front-facing tool for people to actually use to configure something. What I at least am thinking of is the [inaudible] slave name server actually zone transfers the content of a new zone from its master, and that's where the protocol really comes in where slaves that do not know about certain zone can still go to what the provisioning master would be and get configuration of the new zone.

So I think we're talking about two different things there. We have the ease of configurability with tools that might have a HTTP-rest interface for [inaudible] JASON [inaudible] for all I care, whatever. It's not the API there that is important, but it's the ease of use by people. But what is important is how name servers talk to each other in an automated way – in a secure automated way – to get what they need to actually provision up in themselves or push forward to other slaves, and that's what we've been doing some work.

We're actually [inaudible] so you have some name servers that don't know anything about each other accept for their secure shared secrets and they will accept from each other anything in terms of configuration, which makes sense, which is signed properly, and that's just name servers talking to each other like now they're doing through the zone transfers, but that would go a bit further on a meta level where you actually zone transfer a meta zone, which is a configuration of zone.

So I don't think we're contradicting each other, but I think we're working on two different things here. You're working on the user interface, which makes sense. We're working on the back ends where interoperability again makes a lot sense [inaudible]. Thank you.

UNIDENTIFIED MALE: I don't want to contradict. It's indeed true. One of the things that we wanted to have – I think we all want to have – is someone say, "Look, this server is going to be a slave, is going to be listened to that one configuration out there, and when we change that configuration, when we add the zone over here, it also gets added over here." I think we're in full agreement on that, and we also do that.

Where we get into irreconcilable problems, getting back to your question, is when we differ in opinion on semantics. For example, if one implementation says, "Look, we can have this zone but it's frozen, so we have it but it's not being served, and then it needs to be engaged," or whatever, if other servers do not share those semantics, they will have a rough time emulating that stuff.

So that's one answer why it's hard, but I'm in full agreement that we do not – and also our API is not a user interface API. It just so happens that our user interface is using our API. But it's not limited to that, so you can ask it, "Give me a list of all zones/remove all zones," all stuff. But there clearly is work to be done.

BRIAN REID: I'm aware of four different profit-making companies whose business is layering a UI on top of BIND. All of them do well. All of them have customers who really like what they really do. We stay out of it, and what's startling to me is how incompatible their four different UIs are.

They're all the driving same software underneath. All four of those companies have maintenance contracts with us, but two of them have more engineers than we do.

So I don't think UI is part of a name server, and I'm very happy that people have UIs for their name servers, but it's a different business. It takes different skills, and in my opinion and ISC's opinion, it doesn't really belong in the core product.

ANDREW SULLIVAN: So that's an interesting position. I don't necessarily disagree, although I think we see in the DNS world that all these servers up here have – the very core functionality is similar, but the different approaches to DNS signing, the different approach to splitting out resolvers or not, all the different capabilities you could arguably say should or should not be part of your DNS software.

I would also like to note that while ISC sees a case, whether it's big companies that have existing provisioning systems and are happy to shoehorn in BIND 9 or whatever they're using into their framework. I think that's probably not the interesting problem. They'll come up with a solution somehow, no matter what they get presented with.

I think the more interesting problem is how do we make it a company that doesn't have a team of 100 engineers running a DNS, give them something they can put into a provisioning framework or something like that.

I think really the only people that really do that now are Microsoft, right? Honestly. But I don't know. Is it a problem that there's so much diversity in this space?

UNIDENTIFIED MALE: Good.

UNIDENTIFIED MALE: Let me ask this question slightly differently to pick up on something that I forget who it was, but somebody said during one of the presentations to get away from doing this all with VI, and the reason for that is people are losing – it's a set of skills that people are losing.

It seems to me that part of the issue here is that the DNS never really had a distinction between the control and data planes, and so on the DNS protocol people of course always want to put everything into the DNS because we only have one channel, and every application designer in the world wants to do this by having some sort of control plane which is separate from the data plane because that's how you design applications if you've got half a brain.

So we have this sort of deep tension, and I wonder if maybe the difficulty is that we really have an application here in this case that we're talking about – the DNS – and we're not designing things according to good application practice. I think that may be something along the lines of what you were saying.

UNIDENTIFIED MALE: I think if you have only one zone like the average gTLD or ccTLD operator has, VI is fine because you one time added your file and you never, ever touch it again, while if you're were hosted and had like 100 to 1000 domain names coming in and leaving your systems every day, VI is not fine anymore because it's basically impossible to do it in a manual way, independent if it's VI or [inaudible] or any other editor that you might think of [note but] to say something else.

So you have to have some automatable way of provisioning that, and that's what the provisioning is all about. Then again, if you think that different name server implementations are important, then you need to have something where you don't have to implement a provisioning mechanism for this name server, another one for this server.

I think ISC and BIND is not actually saying provisioning is not important. They are. But what they're saying is the only way that thou shalt configure the name server is by putting a conflict file again the name server itself via any mechanism that you choose, and then tell the name server to load it, parse it and make it happen. So they actually take a stance on how provisioning works. That's the way that BIND works in terms of provisioning.

BRIAN REID: No, it isn't not. You can configure BIND with RNDC, and you can configure BIND with dynamic updates. So that was true five years ago. It's not true now.

ANDREW SULLIVAN: I see we have a line, so we will go to the floor.

UNIDENTIFIED MALE: [inaudible] Labs, but also partly engaged in – yeah, yeah I'm still in [inaudible] Labs. But also engaged in IETF business, and I've heard somebody say it doesn't work in the IETF. In fact, if people wanted to work in the IETF and they come together with the drive to get something done, usually things work in the IETF.

What I observed at this table is a certain drive, but in slightly different directions, and if it doesn't work at this table, it will certainly not work in the IETF.

So to the point that you were making – why is the IETF not functional in DNS area – I beg to differ a bit here. But it's important, if you guys go to the bar, there's 90 minutes in which you don't do anything but cheer for the orange team. Then after that, you're in the right mood and get together and you agree that this is actually useful for a bunch of people that actually operate name servers and not want to have this diversity. I guess if you write the proposal, you write it down, you go to the IETF, the decision to implement is yours at the end. I think if you go together, there is already a good basis for a rough consensus. It should be easy.

SHANE KERR: I was the one who said it was hard, and I say it was hard because I was involved with the last time we went to the IETF. We went and we did the requirement problem statement like you're supposed to do, and went to the area directors like you're supposed to do, and we asked to boot up a new working group and they said no, and we said, "Okay," and it got thrown into another working group and then died quietly on the vine.

So maybe we didn't do it right, but it certainly doesn't work the way it's supposed to.

UNIDENTIFIED MALE: [inaudible]

ANDREW SULLIVAN: I don't know.

SUZANNE WOLF: Suzanne Wolf, speaking with no particular affiliation. Shane, I'm glad you said that about trying to make the IETF process work because the other piece of it is for any standardization effort, for any way for people to work together, there is a set of processes. There is some navigation. There is some discussion. And the other piece you have to have is people who are willing to say, "We'll help you with the process. We'll work on making that stuff happen – move the furniture so the work can get done," if the will to do it is there for the work.

I think you have at least one friendly working group chair here. That's why I took the gig. So see me after class.

UNIDENTIFIED MALE: [inaudible]

DAN YORK: I would just partly commend you all for being here, and this was great. I got a lot good notes out of it this and some pieces right here.

I would second this thought that if we can look at how we can figure out some common provisioning, at least across the ones that want to participate, because of the fact that you guys know I do a lot on the DNSSEC side. One of the barriers we have on the DNSSEC side is challenges with provisioning. That's part of the pieces that are there and among them. So it would be excellent for that component.

And Shane, I think, too, it speaks to what Suzanne just said. Yeah, IETF has cycles and different pieces there. Right now, certainly within DNS-OARC, there's a lot of interest around these deployment issue, and we're certainly that with DNSSEC-related stuff, again because I think people are saying now we wrote all these nice standards. They're not getting out there. What's wrong? So people are looking at there are these deployment-related issues. So we're seeing more interest in that.

I think it might be a time again to try to run that up there again and see what happens. I'd be willing to help, certainly. I think others here would, too.

ANDREW SULIVAN: I'm impressed and delighted that we have managed to spend so much time on the topic of provisioning without coming to an agreement about how we're going to fix it, but I wonder if there are other topics that are gnawing at people. You've got your chance here. Take the hits while you can.

UNIDENTIFIED MALE: Is this on?

UNIDENTIFIED MALE: Yes.

UNIDENTIFIED MALE: [inaudible] again. In your introduction, you raised an interesting question. We see a lot of co-diversity. One of the questions you asked is, “Are there dependencies that you share?” Obviously there might be dependencies in, say, event libraries, in SSL libraries, and what have you.

I’m sure all the software has bugs, but they will be different if the code bases are different enough.

You also touched upon everybody looking at the same reference code, which is more a protocol-error type of consideration. Now my question to you as a moderator for this channel is, why did you ask that? That’s not completely clear to me. Then maybe then the panel could answer a few questions particular on that side of looking at reference implementations and where there might be problems in that, because I’m sort of interested in that question there.

UNIDENTIFIED MALE: The reason I raised it is I know a little bit about the DNS protocol and I’ve read I think most of the RFCs probably more than once, and yet I continue to find that series of documents that is supposed to define what our protocol is like to be a little mystifying.

I have to do work every time somebody asks me a question. I'm in the office and people ask, "Well, why is it this way?" and then I have to go and like pick up one of those threads that I've left for myself and then trace it all the way back in order to find our why that is. And I've been working on this for a number of years.

And when we have a sticky problem with how should this work, what do people do? They all go the same mailing list. We all talk to the same people, and we say, "How should this work?" and then we come up with some sort of agreement.

What I worry about it is, sure, okay, we've got this code base, but it's written by a bunch of people who are all essentially thinking the same thoughts all the time. When you have that kind of commonality of the single community, the danger is that they're all going to implement the same bug because they've all got the same introductions. I just don't know whether that's true, but it's the kind of thing that keeps me up at night.

UNIDENTIFIED MALE: Please, go ahead.

MEHMET AKCIN: I want to follow up on that because one of the reasons when I first joined Microsoft and was thinking about actually, "Hey, I can just implement NSD and I'll be done," I looked at Windows and when I spent time with these guys and understood the mentality behind delivering functionality and delivering speed and performance and the capability, it was totally different. They were trying to solve a different problem than what I had always tried to solve with actually the same software.

So I think diversity is important in that sense, not necessarily sharing an operating system, but at the same time, even though there are operating system differences, if it is the same people behind writing the code, I think we should [inaudible] whether it needs to be diversified as well.

Some vendors out there that are not necessarily sitting on this side, but on that side is doing that greatly as well in different operating systems.

ANDREW SULLIVAN: I'd just like to say that, yes, indeed, there is [convergent] evolution. People do create the same bugs in their code. I'm trying to remember the specific example. I believe it was something with the zone transfer code with tsig or something that someone discovered a bug that ended up being a bug in all the known open source versions. It just happened to be that we all implemented slightly different buggy versions of the same basically unused feature. But there's no reason to think that it couldn't be a very highly used feature that just no one had happened to trip that code. So it does happen. People think in certain patterns and hackers think in the other way and break stuff.

MEHMET AKCIN: Just something real quick I forget to say. When we developed the RFCs, when we tried to implement the RFCs within those DNS, these [inaudible] any open source. It's not because actually we cannot, but it's really because we don't want to just copycat the same thing. It's not fair. And we want to deliver something much more diverse and available for people. This is a practice that, as far as I know, has been the practice for Microsoft for many years.

OLAF KOLKMAN: But that is independent implementation of code, and I think what you were looking at is independent implementation of behavior, which is slightly different than independent implementation of code.

ANDREW SULLIVAN: Yes. So that was Olaf and this is Andrew again, and I'm just saying that because I guess the poor folks online can't tell by our voices. The thing that troubles me about this is that on the one hand, the whole point of having RFCs is that you're supposed to be able to look at them and different people can implement them and come up with the same result, because obviously we want interoperability, in fact.

I don't know about other protocols – well, I do – and in other protocol areas, actually, this is a little bit better, although not a whole lot. But in long-lived protocols – the DNS is one, the SMTP is another example – there are a bunch of people that you have to ask in order to say, "What do you do here because I don't understand this part of the protocol?"

However, the DNS community has been much worse than most of the other protocols in keeping their documents up to date. So for instance, SMTP has been through several revisions. I'm thinking of it because it's just as old. It's been through several revisions in order to keep things up to date and to try to clarify stuff. The last time the DNS world did a serious clarification document was 2181.

Paul Hoffman is working on a test suite that is trying to look over what conformance means for different kinds of DNS implementations, and he's currently got a list of 100 RFCs that you have to conform with in order to do the full thing.

Well, I don't know about you, but I am suspicious that the DNS resolver code that is in a home gateway device that all of the customers of every ccTLD and gTLD in the world are using in their houses, I suspect the engineers working on that are not going to read 100 RFCs. Maybe I'm wrong.

But I think what will happen is they'll read a couple, they'll look at what one or two servers have done, they'll look at the code – probably they'll look at BIND because that's the one they've heard of – and then they'll implement that. I wonder what that does to us as operators of high-level infrastructure on the Internet. What kinds of consequences does that have for us? I think that's the reason I was asking that question. I don't know if any of the panel wants to comment on it.

OLAF KOLKMAN: Maybe a small comment. I think that in practice there's quite a lot of diversity, also in ways of thinking. There are typical ISC ways of solving a problem. There are typical NLnet Labs ways of solving a problem, and there are typical PowerDNS, and Yadifa, and Knot ways of solving a problem.

In fact, some of the diversity we have seen has saved us already. For example, the different resolvers have different retry behaviors in case authoritative servers are down. Some servers are far more stubborn in keeping on asking the same servers, and others are not. This diversity has made sure that some botnet attacks are not universally effective because they disable one server, but they don't disable the other, and next time it's the other way around.

So I think that maybe as the reverse of what was said previously, we don't standardize all our behavior behind this table, and I actually think that we are diverse enough that this is only rarely a problem. This is mostly also, by the way, because the RFC specifications, although there are indeed around 100 of them. If you follow all the trails, they leave quite a lot to the imagination, and this has resulted in sort of natural diversity.

ANDREW SULLIVAN: Okay, so we have two people and one minute, so very quickly, Brian and then [Mattias].

BRIAN REID: Diversity is hard to achieve. I offer as an example the Kaminski bug. The programmers who implemented that all failed to think about port randomization on a reply and it—

OLAF KOLKMAN: I did not fail to think about that. The RFC we wrote to mandate that was effectively blocked by people who did not want to think about it because they wanted to further DNSSEC, which is my pet peeve by the way, which you might have realized.

UNIDENTIFIED MALE: [inaudible]

OLAF KOLKMAN: I'm very serious, yes.

[MATTIAS]: One small comment first. At the last clarification, RFC number was 6841, which was a DNSSEC clarification. But overall actually, I do agree with you, Andrew, and I also agree that some RFCs can be very ambiguous if you're talking about interpretation. I specifically recall things about signing with different algorithms.

On the other hand, maybe on a good note is that once there is an error in the protocol description, everybody did interpret that correctly, and everybody had that error, which was a – I'm referring to a case when there was a missing NSEC 3 record on a no data, which is above the unsigned delegation, which is a very, very rare corner case. But luckily enough specification in that document was clear enough that everybody interpreted that error and made that error.

ANDREW SULLIVAN: All right, we are officially out of time, but I saw you waving.

UNIDENTIFIED MALE: No [inaudible]

| | |
|---|---|
| ANDREW SULLVAN: | Oh, to the closer. So since we're officially out of time, I want to thank all of the participants on the panel today, and I want to thank all of you also for your remarks and questions. Thank you, everyone. |
| EBERHARD LISSE: | Okay, so we're not done yet. Warren, you can sit down again. Norm is going to close as usual with a few remarks. |
| NORM RITCHIE: | Thank you. Eberhard asked me to close up with a few remarks. I don't know why. I guess nobody else would take the job. |
| UNIDENTIFIED MALE: | [inaudible] |
| NORM RITCHIE: | Oh, it's rotation. It's difficult to close this. I'll just give my perceptions of the day, so like a review. These are my notes that I made myself during the day. Overall, this is the largest Tech Day ever. I haven't taken the official numbers, but it's pretty obvious just looking at the audience, 150+. It's also the largest ICANN apparently, 3300. But that's great. I also noticed something different here. I've been to every one of these I think since the dawn of time. It's the audience is more diverse now. It used to be ccTLD tech people, and that was it. It was like having group hugs all the time. But I think we're a lot richer for having the different diversity and gTLDs here now and some non-technical people. I think |

that's a great sign if cyber-security – actually that's me now – I think that's great.

I found the presentations today very entertaining, interesting. They were all well-presented and very timely topics. So I thought it was great. So kudos to the presenters and thank you. And also kudos for you for organizing the agenda, Eberhard. I know you worked very hard on it.

Just my notes on the day to get a recap. The label generation rule set from Kim Davies, I thought that was very interesting and very much needed. I don't know how many times I've been asked in the last three months, "How do you validate a domain name now?" With the new gTLDs coming out – what's that called? Te ones the browsers use – no, no, no – the lists that the browsers use.

UNIDENTIFIED MALE: Public suffix list.

NORM RITCHIE: Public suffix list, yeah. So that's not quite up to date all the time. I've heard complaints about that. Some ccTLDs are changing their own policies. UK is a great example. You have IDN domain names as IDN TLDs. So a lot of people are having difficulty doing that, so I hope Kim actually extends the work he's doing now and actually gets into something that's like the public suffix list.

CentralNIC is again talking about then woes in dealing with the DNSSEC and real time zone updates. I think we heard that many, many times and it still continues.

The DNS analytics, love that stuff. As someone said, this is porn for DNS people. I agree with that. I thought it was really interesting, Eberhard, what you've done with a smaller registry and limited resources.

Then you have Nominet – tons of resources. I know that they were working on it for a long time and they've just done amazing things. There's a lot of intel in there that they've only touched the surface of today. I really hope you can convince Roy to give a longer presentation on that, and I hope that he will share that with us. Maybe something for LA would be awesome.

IETF update as always, it's great to have that. I really appreciate it, because like many people here I think, don't go to IETF. But I was a bit surprised that one topic that was presented was persistent surveillance, and that was it. So my take away from that is that must be a very high priority for the IETF.

UNIDENTIFIED MALE: [inaudible]

NORM RITCHIE: Oh, [unless you're] asked you to present? Well, that's the other reason.

Then the next IETF I think is in Toronto in July, so I might go to it since I'm actually not far from there. I can drive there.

For those of you that may be thinking of going, Toronto is a great place in July. It's a great place all the time, but in July it's very nice there.

The newly-registered domains, I think I'll follow up on that, although I kind of share that concern. As I mentioned, I do cyber-security now. Not all new domains are bad, so I think there's a better way of approaching that. I have some ideas on it, and I look forward to working to correct that.

The registry locking, a simple and elegant solution. Very well-done. I cannot see any reason where a registry does not offer that. If you're a registry and you're not offering it, rethink it. It's a great service. Everybody should have it.

Then the DNS Round Table, finally. Fresh in everyone's mind. I won't repeat it. My own personal take away on that, I used to be the CIO at CIRA.ca. So the DNS has to follow certain standards. It has to be interoperate, so therefore they're all kind of the same from that regard. You have choices. Oh, the other choice not presented was outsourced. [You also get] a DNS service provider.

So do your homework. Select what works for your needs, your budget, your resources, and your in-house skill set, because that's really what the big differentiation is going to be.

And that's it. Thank you very much. Turn it back to you.

EBERHARD LISSE: Thank you very much. That actually closes it for LA. As I said, we have a joint meeting with DNS-OARC, and we're going to try to look – Jaques Latour actually from CIRA suggested it – that we look deeper into data.

I think we close the proceedings here. Thank you.

**[END OF TRANSCRIPTION]**