KOBE – Tech Day (3 of 3)
Monday, March 11, 2019 – 15:15 to 16:45 JST
ICANN64 | Kobe, Japan

EBERHARD LISSE: Okay. Can everybody settle down and sit down, please? Or the leave the room. Then we can start with our next presentation. I don't want to mention people by name, but some names I know.

UNIDENTIFIED MALE: Hi, Warren.

EBERHARD LISSE: Hi, Warren.

UNIDENTIFIED FEMALE: There's something wrong with the [inaudible].

EBERHARD LISSE: Don't worry. Put your color glasses on.

All right. The next one is John Levine with the second presentation on how to make your mail EAI compatible.

JOHN LEVINE: Thank you. I'm a completely different person from the one who talked about IDNs this morning, and this is a different topic. As always, I qualify my audience. How many people are somewhat familiar with EAI mail?

Eh, a few. How many people have ever sent or received an EAI mail message?

Okay. A few of us.

UNIDENTIFIED MALE: [How many are opposed to spam?]

JOHN LEVINE: I'll send you another one just to get your denominator up.

So, come on … whoops. There we go.

Okay. Yes, remind me to – well, it's an acronym like ISO. So here is my dandy new address. It has accented letters. It has non-ASCII letters both on the left and the right side of the address. This is one I just use for testing, but it is a real address. If you send me mail, I will probably get it and … well, give or take the usual spam filtering.

So here is a brief history of e-mail. So, back in the good old days, all the e-mail was ASCII. So Boris wrote to Ines and this message was entirely in ASCII. This was the mail looked back in the 1980s.

Then, in the 1990s, we invented MIME, which allowed us to put non-ASCII characters pretty much everywhere except the addresses. So we can see that Boris' name can be written in Cyrillic in the comment part of the From line. Ines' name has the tilde over the n where it belongs. The subject is in badly translated Russian, in this case, and there's an accent there in the text.

MIME was invented in the 1990s, and this was pretty widely implemented throughout Internet software, I would say, by the early 2000s. And we sat there for quite a while.

So EAI, which has just been finished certainly within the past decade, maybe the past five years, completes this. So now the e-mail addresses can have UTFA characters. So here's Boris' name in Cyrillic and here's Ines' name with the tilde where it belongs again.

So basically EAI let's us complete the process. I think it's particularly important for people in China, people in South Asia, where there is large linguistic communities who don't have a lot of experience with English. So it really is important for them to be able to use their own languages.

So here's what we're going to do. How many people are familiar with the guts of SMTP? Sort of?

Okay. Most of us. I'm going to skip over the Unicode and IDN part pretty quickly because this is ICANN and I hope you already know about that.

So here's a domain name, and here's the way your mail system works. It is conceptually divided into these four parts. The sender uses a mail user agent, an MUA, to compose the message, which is then sent off to the sender's mail system, the mail transfer agent, which then figures out where the receiver is, sends it through the Internet – that's that cloud in the middle – through the receiver's MTA, which then holds onto it until the receiver is ready to get. Then here she is reading at her MUA.

Every mail message takes this conceptual path. In some cases, the pieces are integrated together. In particular, if you are using web mail, your MUA is implemented in your web browser, and the MTA is the mail system that your web browser is talking through. Behind the scenes, this always happens.

Then to add some more acronyms here, the process of sending the initial message from the person who composed it to the initial MTA is called submission, which is a little bit different from the standard transmission because, in the process of submitting it, they tend to typically e-verify who the sender is and you clean up minor technical errors in the header. You add missing dates and add missing messages IDs and stuff like that.

The protocol back and forth the sender's MTA and the recipient's MTA is SMTP. Once the recipient's MTA receives it, it then holds onto it, and then the recipient's own computer typically retrievers it by either POP or IMAP, which are the old and new ways to retrieve mail that's been received. Or, again, it might be web mail, in which case it's all integrated together.

So here under the cover is a sample SMTP session. So when Boris sent that message to Ines, Boris' mail server and Ines' mail server had this conversation. In each line, R stands for receiver and S stands for Sender. So the sender finds the recipient's mail server, which is the DNS part of this that uses MX records. The recipient's mail server says, "Okay. 220. I'm here." The sender's says, "Hello. I want to have a mail session." The recipient then writes back and says, "Okay. I'm here, too. And here's the list of SMTP features that I can handle." The only one it lists here is 8BITMIME, which basically means that, if you have Unicode in the body of your message, you don't have to do any special encoding. You can just send it as is.

The next one is, "Okay. I have mail from Boris and it says okay." I'm sending it to Ines, and it says, "Okay." There can be multiple recipients, although there isn't in this message. Then the sender says. "Data. Okay," and the recipient says, "Okay. Send your message." Then it sends the whole message as one big block of

text – the From and the To and all the headers – and then followed by the contents of the message.

At the end, it sends a line with a dot to say it's done, at which point the receiving system thinks about it for a moment and says, "Okay. I accepted it. I'll do something with it." Then the sender says, "Quit," and they're done.

So this process happens for every mail message. So there's the hello, identify the From and the To, and you're done.

Now, again, this is another part I can go through quickly. Traditional mail is encoded in ASCII, which is a familiar 7-bit character set that I hope we're all familiar with. But unless you happen to speak either English or Hawaiian, ASCII does not have all the letters you need to write text in your language.

So writing systems use scripts. Most systems use a single script. Russian uses Cyrillic. Here in Japan, Japanese uses three scripts: Hiragana, Katakana, and Kanji. It's also fairly common to mix in Latin script as well.

So Unicode attempts to solve the problem of how do you encode everything that's not ASCII. There have been a whole bunch of different efforts over the years, which I will not talk about since by now they've all pretty much gone away in favor of Unicode. Unicode has a bunch of advantages over its predecessors.

The main advantage is that in Unicode every character has a unique representation. There's no shift codes or anything like that.

So the goal of Unicode is to be able to represent every possible character in every language that anybody speaks. It doesn't necessarily represent it uniquely. Unicode is by design sort of a typesetting language. The goal is that there is some combination of Unicode glyphs that will allow you to write whatever it is you want to write, although it may also be there's more than one way to do it. This is totally standard in typesetting. Back when we were setting type with cold type, we'd frequently have two or three different ways to compose the same letter, which looked fine on the page. From Unicode's point of view, that's fine, too. It's not so fine for computers, though.

So here are some typical Unicode code points from different alphabets. So there's a Cyrillic letter, Hiragana, and Arabic. Then the last three is – here's the Roman letter A with an accent. That's Unicode code E1. But you can get something that's visually the same by entering a regular ASCII A, followed by this acute accent, which is then displayed on top of the A.

But, again, every Unicode code point has a code, and they're written like this: U + number.

Now, computers use bytes and Unicode values. Since there's a million Unicode values, you can't encode a million different codes into a single 8-bit byte.

So, this ASCII, it was easy. ASCII is only 128 characters. That easily fits into a byte. In Unicode, it doesn't fit into a byte. So the way we typically encode it is UTF-8, which is a variable length in coding. The first 128 Unicode characters are the ASCII character set. So if it's an ASCII character, the Unicode is the same, which makes things a lot easier for us when we're programming. If it's something else, it's some longer combination of bytes. It's anywhere from two bytes to four bytes. They've made some attempt to make the more common code points have shorter encodings, but not really.

So with UTF-8, we have a way to encode all of the text we're going to do, but I want to reiterate the fact that it's variable length, which causes extra complications. If you have an ASCII string that's four letters, you know it's four bytes. If you have a Unicode string that's four characters, it might be four bytes or it might be 15, or if the characters are actually composed of multiple items, it might turn out to be 20 or 30 bytes. So the length of the UTF-8 you know will be as much as the characters but potentially quite a few more.

So in e-mail addresses, you can have IDNs. We all know what IDNs, are, right? What we did this morning. Okay. So e-mail addresses with non-ASCII characters are internationalized e-mail addresses, which are abbreviated as EAI because that was the name of the IETF working group that standardized this stuff.

So, once again, I'm reiterating that there are A-labels and that there are U-labels. The A-labels are actually used technically in the DNS. When I was talking this morning about the two million IDNs, all of the IDNs that we're looking at were represented in the DNS as A-labels, like this xn -- stuff.

Human beings can't deal very well with those, so every A-label has a Unicode equivalent, a U-label. So this Chinese string, which, if I recall, means hard-to-understand string, or something like that, translates into this very long chunk of A-label.

Now, an e-mail address has two parts, and the two parts were handled quite differently. There's the local part, which is before the add sign, and there's the domain part, which is after the add sign. In any EAI address, both parts can be Unicode. The domain part is easy. The domain part is an IDN. The second path of an e-mail address has always been a domain name and it continues to be a domain name and the only difference is there now can be a U-label as well as an ASCII domain.

So now this slide is the meat of the talk, which is, well, what do we do about the fact that we have old-fashioned ASCII mail systems that only handle ASCII mail, and we have nice new EAI mail systems that can handle EAI mail? What sort of compatibility is there between them?

The answer is the IETF spent – what would you say? – maybe a decade trying to figure out ways to do automatic backward compatibility and downgrading and all that stuff. Did it work? No. It did not work at all.

So what we have done here is EAI mail is conceptually a new mail system, although in fact it's running on all the same mail servers and networks.

But the point of these two thick green lines is, if I have an ASCII message and I'm sending it to an ASCII recipient – that's the upper green line – that's traditional mail that works the way it always has. If I have an EAI message which I'm sending from an EAI address to an EAI recipient, that's conceptually this new system. They're both EAI so they both work.

Okay. The two diagonal lines there. If I have an ASCII message and I'm sending it to an EAI recipient, that always works since ASCII mail is a strict subset of EAI mail. So if I'm Bob and you have an EAI-enabled mail system, you can always send me a message.

However, if there's an EAI message and it's going back to an ASCII recipient, in general that doesn't work.

There are a few narrow cases where you might kind of be able to make it work, but in general, if you can't start out with ASCII addresses, you can't expect an ASCII system to receive it. This has been the biggest problem for implementing EAI mail. Until there's a sufficient number of EAI mail systems, you don't have anybody to – it's sort of like being the only person with a telephone. Having one telephone is not useful. Having two telephones is much more useful. And having a million telephones is way more useful.

So we characterize EAI support into two levels, which we creatively have named Level 1 and Level 2. Level 1 is a system where your mail system only has ASCII addresses but you have enough EAI support that you can receive mail from EAI addresses and you can send mail to EAI addresses.

It turns out there are a lot of these Level 1 systems. In particular, Google's Gmail and all of the domains they host there is all Level 1 compatible. Microsoft's Outlook is Level 1 compatible. And there's a few other hosted mail systems. But just between Gmail and Outlook, that's a significant subset of all the mail in the world. So if you're Gmail or Outlook, you have an ASCII address, but you can correspond with anybody.

Level 2 finishes the job and you start assigning EAI address on your own system, which turns out to be much harder.

Okay. So the hard parts about Level 2. The first part is assigning good addresses, where good means easy to remember and easy to type and relatively difficult to screw up. It means matching addresses and incoming mail, which I will talk about in more detail. Remember, Unicode is a type-setting language. There's frequently multiple ways to type the same thing. So you'd really like it that, even if there are 14 different ways to type your address, they all end up in your mailbox.

And for the foreseeable, we will still need kluges for ASCII compatibility. Even if you're a Level 2 EAI mail system, unless you happen to be living on an island where everybody speaks the same non-ASCII language, you're going to need to correspond with ASCII-only mail systems.

So, harking back to the – there we go – two green bars, we made changes to SMTP both so that the recipient mail system says, "Yes, I can receive EAI mail," and so that a sending system can say, "Yes, this message is an EAI message." They need to be separate because even a system that handles incoming EAI mail might not have every single address handling EAI mail.

So the differences are that – here's more or less the same mail conversation but now I have annotated it for EAI. So the first thing

is that, when the sender connects to the recipient system and says hello, the recipient system, in its list of features it handles, also says, "I handle SMTPUTF8. I can handle incoming EAI mail."

Then, whenever the sender is sending a message, if the message is an EAI message, it puts this SMTPUTF8 tag on the end of it. So it says, "Okay. Here is a message from" – and I'm sorry my Chinese is not good enough to pronounce that name – "whoever this is." Oh, I remember. It's Elvis. "Here's a message from Elvis and it's an EAI message."

Beyond that, the mail protocol is – yeah. I didn't show you the rest of this because the rest of it is unchanged. The receipt to is unchanged, other than it can have a UTF-8 address. And then the data and the message is sent is exactly the same way, except now the message might have UTF-8, where it only used to have ASCII.

Okay. So if you're writing a mail server – I actually haven't actually written a mail server, but I've certainly patched and debugged mail servers, so I can tell you that I've done some of this – you need to make your mail server advertised, "Yes, I can do SMTPUTF8." When you are sending mail to other people, you need to check for that and make sure that, when you have an EAI message – and on your mail system you'll invariably mark some of your message as EAI; some of them are still pure ASCII – you

need to set the tag and you need to make sure that the other system says it's okay.

It's possible you'll see mail from as SMTPUTF8 and the other system will send you back an error code that says, "Gee, I'm sorry, but this address doesn't handle EAI mail." You need to be able to handle that error and reflect it back at the user somehow.

In the user – yeah. The standard says that the domain names in the addresses are supposed to be U-labels. The reality is they'll be a mix of U-labels and A-labels so that you should accept both of them and handle them correctly, which means, within the mail, they are EAI. They're all U-labels. When you're looking up addresses in the DNS, you got to turn them into A-labels. It's not hard. There's libraries to do it. You just have to do it.

Finally, there's this thing I mentioned before and I have a whole slide on later about fuzzy matching. When the message comes in, the local part may not be exactly the same local part you have on file. But frequently it'll be close enough that you can basically automatically correct it. For example, if the address has an accent and the user left off the accent, you should deliver it anyway.

So once the message has been received, then the recipient user will need to pick this up. But these slides will all be available. No need to take pictures of them. I can give you a copy.

So we've upgraded POP3, so it now has the UTF-8 option so that the user names and passwords and text strings can all – remember, when you pick up your mail, you need to log in. So the user names can now be UTF-8 and the passwords can contain UTF-8.

IMAP, which is the more complicated system that is used more commonly, also has options for UTF-8 user names and passwords. IMAP can search through the mail box to find this string, and it has actually too many options to do UTF-8 strings in search strings and folder names.

Both of them have what we call a downgrade option. The problem is this. You have a whole UTF-8 mail system and you've received all this mail for your user and your user logs in from an old mail client that doesn't handle UTF-8.

So you have the problem that your mail system says, "I have this mail system for you, but since you're not a UTF-8 mail system, I'm not going to give it to you." That seemed gratuitously cruel, so the IETF defines some optional ways that you can basically just whomp the messages into an ASCII approximation so that, even though they can't [inaudible] properly, at least they can see the message and they can see who it's from and do something with it.

POP and IMAP support have been lagging. At this point, the open source mail server I know of that handles EAI is Courier, which is a fairly nice system written by a guy I know in New York. It's not super popular but it's totally free and it works pretty well. If you want to experiment with EAI, it's worth a look. It's pretty easy to set up.

Actually, since we have Google people here, I know that Gmail and Outlook provide IMAP service – I can check my Gmail from an IMAP client. I haven't checked to see whether Gmail's IMAP server handles UTF-8. Anybody know?

Yeah. Well, it should. I presume if they don't they will soon.

So anyway, the mail user agent, the thing the user actually uses, is the obvious stuff, like, "We'll handle Unicode names." But also remember you're going to have Unicode addresses in your address books, and you're probably going to have Unicode user names to log into the mail server and Unicode passwords.

The domain names. Now that U-labels are valid, validating the domain name is allowable [and] an e-mail message has changed a little bit. So there's ways to do that.

Again, the same extensions I mentioned for SMTP also apply to submission. So it's also possible you could have an EAI-capable mail client try to submit a message and then your mail transfer

agent will say, “Sorry. [inaudible] address. Can’t accept EAI.” So, again, the client’s [offer] needs to be able to understand that error message and do something about it.

This is fairly obvious, but while you’re internationalizing it, the headings and the prompts and the subs should all be in the local language, too, meaning a lot of mail programs already do this but the idea of a mail program where the address can be in Chinese but it still says [daith] in the English at the top of the column seems like a bad idea. So fix that too.

A few things. If you’re assigning your own e-mail addresses, you ought to avoid addresses that can confuse users. The IETF has some stuff called [praycee] or string prep. The Unicode Consortium has some script rules, and it’s all fairly obvious. If it’s a Cyrillic address, it should be Cyrillic. It shouldn’t be some combination of Cyrillic and Arabic and Chinese and [Debingari] because, even though that is technically valid, it makes no sense and nobody could type it.

Also, the address space of mailboxes is really huge. So even if you are in a country where Bob and Bob with an accent and Bob with a slash are different vowels, don’t assign all three of them. Just assign one of them and make the other two deliver to it because, even though people who speak language know that they’re different, we Anglophones don’t have a clue. So even if the

address is supposed be Bob with a slash, I'm going to type Bob with a regular o. See, might as well make that work.

Okay. And that's more of this fuzzy matching. Technically, upper and lower case e-mail addresses are different, although in reality we treat them the same. That's because the incoming mail server allows upper and lower case to be equivalent. In the same way that you allow ASCII up and lower case to be the same, you should allow all these minor variations that I was talking about. The A with an accent? Allow it both as a single character and as two separate characters.

This sort of fuzzy matching is not new. Mail programs have done this for, golly, 30 or 40 years. It's simply a matter of extending it to handle Unicode mailbox names in a reasonable way.

Now, beyond obvious things like allowing the accents to be separate or combined, I can't really tell you what's reasonable because what's reasonable totally depends on the language you're speaking. If you speak Turkish, an i with a dot and I without a dot are different letters, whereas if you speak English, an i without a dot doesn't exist. So you're going to have to have some local knowledge to figure out what mistakes are people likely to make. What are people going to assume are the same? But then map them all together.

I can tell you from experience that the more fuzzy you accept, the happier your user will be because the more the mail they will accept, even from people who slightly mistype their address off a business card.

For ASCII-backwards compatibility, the best thing I can say is, when you're offering EAI addresses for the foreseeable future, you should also an ASCII alias so they can give Anglophones their ASCII address and give their friends their new EAI addresses. They both deliver to the same mailbox. This is sort of a special case of fuzzy matching.

Finally, people keep asking about message downgrading. You cannot turn an EAI address into an ASCII address. You cannot turn an EAI message into an ASCII message. So the only way you can do this is, if the original person writing the message is in her original mail user agent and it says, "*error beep* This EAI address didn't work," and her mail agent knows that she has an alternative ASCII address, she can substitute it there. But that's because it knows specifically about that particular user.

In general, you can't do it. What we discovered after that effort is, rather than trying to come up with workarounds, if you think EAI is worth doing, just support it. Don't try to handle for ongoing breakage.

Finally, here's a few security issues that are worth thinking of. One is that there are a lot of characters that look the same. For example, those three round things. They really are a Latin letter, a Cyrillic letter, and a Greek letter. They all render exactly the same in the font I used here.

So my advice is A) don't assign addresses that only vary in homographs. Do allow [for people] to type them wrong. Just accept them all. We also have longstanding near-homographs, like the digit one in the lower case letter L in the English look pretty much the same.

Okay. Anybody here ever hear of variance?

Yeah, a few of us. Okay. I'm not talking about variant domains. I'm talking about the same idea in the local part, in the local address. If my address is this simplified Chinese address and somebody send the traditional Chinese address, you might as well accept it because it's pretty clear what they have in mind.

So, again, it's the same advice. Don't assign these as separate addresses. It probably would be a good idea to accept them as the same mailbox. In Chinese, it's pretty simple. In other languages, the variants are much more complicated and I don't understand them.

But the variant issue does show up in mailboxes a little bit, but since each mail system controls all of its own addresses, you get to pick reasonable rules, when typically just pick one set of characters to assign your mailbox name and perhaps accept the other was as alternatives when you're receiving mail and don't do anything else.

And there's other ongoing challenges with mail addresses. You can now have very long domain names, like sandvikcoromant. That's an actual TLD. It's much longer than the traditional ones.

There's what I've mentioned. There's two ways to write A with an accent, so you need to be able to handle both of those. Although it's not a very good idea to allow punctuation in your local part, you can. I know a guy whose name is O'Brien, and indeed his e-mail address is O'Brien at whatever it is. On the other hand, is mail provider is smart, so if you also type an OBrien without the apostrophe, it also goes to his mailbox. So, again, punctuation is possible. It's not necessarily a good idea.

I am sad to report that, technically speaking, you are allowed to use emojis in your e-mail address. I don't think it's a very good idea if you actually want to receive mail from people. If your name is Avocado, I guess that's a cute way to write your address, but how are people going to type that?

So, again, try to keep your mail addresses simple and easy to remember.

I think – oh yeah. If you're doing the backend, your logs are probably going to be an A-label, so you're going to need tools to look through and find – somebody said, "What happened to my mail?" Okay. You're going to need tools to be able to search through and figure out this A-label was equivalent to that U-label and that's the one I was looking for. These tools don't exist very much, but they're not very hard to create since the underlying libraries to convert the strings already exist.

So I think that's it. Yeah. So [EAI] software can be hard to debug. I haven't even touched on the issues of left-to-right versus right-to-left, since that mostly affects layout on the screen. If you want to test stuff, you need to exchange mail with a lot of other mail systems. The fact that you can exchange mail with another copy of your own same software only proves it has the same bugs. It doesn't prove that you fix the bugs. So correspond with as many people as you can.

I believe that's it. Do I have a few minutes left?

EBERHARD LISSE: Thank you very much. We have seven minutes for questions.

JOHN LEVINE: Okay.

EBERHARD LISSE: So we can start with the first person at the microphone.

BARRY LEIBA: I just wanted to say one more word on downgrading because, whenever we do these kinds of talks, someone gets up and says, "Downgrading is not really that hard. Here's all you have to do."

JOHN LEVINE: Yeah. To which the answer is, "We tried that and here's why it didn't work."

BARRY LEIBA: Right. We had a multi-year experiment before we put standard versions of these specs. The various combinations of sending to EAI and non-EAI mail and having people reply to all and having the different things going back and forth made it – we tried putting extra mail headers with information for downgrading. None of it worked. Please don't say you have the answer unless you've really tried it out and confirmed that it works.

JOHN LEVINE: Yes. And then I think you'll find that the actual answer is, "Just make it work."

BARRY LEIBA: Right.

UNIDENTIFIED MALE: [inaudible] from JPRS. I'm very negative to fuzzy matching at the server site because it will cause the misdelivery of the e-mail to other people.

So I believe that implementing fuzzy matching is [inaudible]. So the e-mail address has to bind to [your resource] exactly, not using the fuzzy matching but using [inaudible]. There'll be more safety.

JOHN LEVINE: Yeah, you're right. People are definitely going to be implementing this in particular areas. There's a lot of interest in some parts of India, and we've had some interest in China. We've had pretty much no interest in Japan, where everybody handles Romaji well enough that they don't need it.

Yes?

UNIDENTIFIED MALE: Hi. Thanks for the presentation. You mentioned the issues with Unicode e-mails. One of them is the variance. What's the solution for this? For example, in Arabic, we have implemented variance in the domain part. Can we apply the same concept with our name part? What if we can create a unified framework or a unified service that can be used by any mail service provider that can be used and implemented?

I think the current mail service providers will not implement EAI e-mail service, so what do you think about that?

JOHN LEVINE: All this stuff that ICANN has had to do with IDNs is because the DNS does an exact match. So if you're going to handle an IDN domain name, you need to have all these rules about how to prepare it and basically how to take whatever it is the user types and turn it into an A-label so that you can be sure that you come up with exactly the same A-label that is in the DNS.

Okay. E-mail addresses aren't like that because each e-mail address is only interpreted by a single system, by the single mail system that manages the mail, so that this fuzzy matching I mentioned is quite possible.

The reality is that different systems are going to have to have different matching rules because matching rules that make sense

in Arabic don't make sense in Chinese and don't make sense in South Asian languages.

So I think it's going to have to be, to some extent, system by system. There's some very large systems. When Gmail supported it, that was probably a billion users who can already do it.

So, again, this is one of the things that Barry said. If we could have come up with a mechanical way to do it exactly the same way with everybody, we would have done it. But 40 years of experience with e-mail says that e-mail needs to be more flexible than that.

UNIDENTIFIED MALE: Okay. Can we solve these issues that are coming in five years? Do you think we can create a unified framework for the variants for all languages that can be used by Gmail, Postfix, and whatever?

JOHN LEVINE: Possibly, but even if you have two copies of Postfix, they don't have to use the same rules for handling local parts. They have complete flexibility. Just going back and looking at ASCII mail, the mappings I do on my mail server are not the same as somebody does on another mail server, even though he's using the same software.

So local parts? It's tempting to say they look like domain names so we should be able to use the same rules, but in fact, because they are interpreted differently, the way we manage them turns out to be quite different.

EBERHARD LISSE: Okay. One more question?

WERNER STAUB: Werner Staub from CORE. You talked about downgrading. That assumes that we do want to have more than seven-bit ASCII in front of what's to the left of the add sign.

JOHN LEVINE: Yeah.

WENER STAUB: If you give up on that, then you don't have much a problem, do we?

JOHN LEVINE: Oh, no. Yeah, ASCII.A-label works and it works in an ASCII mail system. Certainly there is a strong point of view that says that enough people can type ASCII addresses that it's not a problem. In fact, we went and talked to people in China and it turns out that

the predominant way that people enter Chinese text in China is by typing pinyin. So even though they don't speak English, they know the English alphabet. For this reason, Chinese users are much less, "Yeah, we can deal with it."

Okay. It's different, though, I think, in South Asian languages, where people simply don't know English at all and don't know the English alphabet. Then it's much more of a barrier to them. So it really varies from place to place.

WERNER STAUB: Well, we have [inaudible] [in older country] in the world, and there's no question about that because they need it. Even for public [registry] for their own name, there's just no way to get by anywhere in the world without the ASCII seven-bit alphabet.

But let's say everybody has to [explain that] because as far as nowadays, people cannot actually see the e-mail address. Even on the normal phone, they don't see the e-mail address. They just see the display name. That's the only thing they're given.

So what we're really trying to do by allowing people to have that, if you need a system that is so complicated and requires patching all the mail servers in the world just for that little feature, to me it sounds like saying, if you want bread, we cannot give it to you because we don't have Nutella. We want to have e-mail on

internationalized domain names, but if EAI requires to take the whole thing, if it just needs an e-mail on internationalized domain names, I don't need something in front of the @ sign that is more than seven-bit ASCII. [I can get by].

JOHN LEVINE: Then you don't EAI. Like I said, ASCII.A-label is an ASCII address and that works fine. If that's good enough for you and the people you correspond with, you're right. You don't need EAI.

WERNER STAUB: Well, the problem was that some references, when I tried to do it, sent the Punycode. [inaudible] Punycode. He sent the Unicode, so it crashed.

EBERHARD LISSE: Can you take this offline because we are …

JOHN LEVINE: Okay.

EBERHARD LISSE: Thank you. Jaap Akkerhuis.

JAAP AKKERHUIS: Thank you very much. Jaap Akkerhuis is the next presenter. Use the clicker – ah, okay. Ow, shit! Ooh. Stupid table. And stupid me.

Anyway, yes. Jaap Akkerhuis. What I'm going to do is actually proxy somebody else, and that's Roland from [Reisweig] from the University of Twente and NLnet Labs. He's also listening in and trying to catch a train at the same moment, so we'll see. He might be available for some questions as well.

What is this about? This is about research that's already been taking place for the last five years. That's when people started with the idea of can we measure the global DNS on a daily basis, not just sample but doing it really on a daily basis.

This talk is about why do we want to do that and how it's being done and what you can learn from this. So you'll see. Did somebody try this before is of course the other question.

First, why measure the DNS? The DNS translation – we all know what DNS is. [inaudible] from humans to the machine world and back sometimes. Every network service relies on the DNS – nearly everyone on some form of DNS.

So if you know enough about the DNS you might be able to something of the evolution of the Internet in its protocols. That's the idea. Did somebody try this before? Yeah. There are a lot of people doing similar stuff in a different way.

So this passive DNS is [inaudible] listen to the traffic and try to collect that and make inclusions. [inaudible], you only see what clients you're listening to are asking for. So it's kind of biased, depending on who you're actually listening to.

There's also no idea about how the timing goes. So we need to also do some timing. You really don't know how things are happening. That's why OpenIntel is doing active measurements. They sending a set of queries [inaudible] domains once every 24 hours. That's a lot of queries. So nowadays, there's 260 million domains. That's 260 million queries. There is all the gTLDs available and also I want to see the ccTLDs and see the list of the ones they get access to and the zone files because that's what you need for querying stuff.

Please, if you are interested in cooperating, talk to us. So that's what it is.

So now we are going to do the useful – we always need to do some bingo words, so now I'm going to talk about blockchains or agile or lean or cyber or big data. What do you think this will be? Well, of course it will be big data.

[inaudible] big data is [inaudible] research [inaudible], so that's you get some money from. "We're doing big data and doing a lot of interesting stuff about it." So will this actually be big data. Let's try to measure that before we start.

Well, so we go back to the biology and say human knowledge, say, to the million DNA base pairs. Daily we have about 2.3 [inaudible] DNS records each day. This is three-quarters of humans we are assembling. We've doing that since February. We have collected the genomes of more than 1,000 humans, which is probably more than all the people in this room if you look at it.

EBERHARD LISSE: Can we do a quick headcount, please?

JAAP AKKERHUIS: What?

EBEHARD LISSE: Can we do a quick headcount, please?

JAAP AKKERHUIS: Yes. Okay. So we could mention this as big data. Anyway, how do we measure? How is it going to be measured? We take active measurements, 200 million a day. A lot of people complaining about being probed all the time and some other measurements people are actually complaining about. So [inaudible] stuff. But the way the OpenIntel people is they actually make sure that it always come from the same source, so if you want to block them, it's easy to do that. It's well-known. They've also documented

what it is, where it is, and who is responsible in it in the RIPE database. They have actually reached to big operators in the data set for whether or not they see this problem happening or have [inaudible], and there was no problem at all. The actual amount of complaints is less than five over five years, so this is quite a careful matter without upsetting too much people in the world.

This little weird picture at the bottom shows you the creation, the first day of the 24 hours. So [there] is some matters in that. But it continuously [inaudible] 24 hours.

Now, what can you do with this data? Well, there's a lot of stuff. So we'll give three examples. One is about DNSSEC operation of practices. Can we say something about it? The other one is about DNS science might be or is improving. Then there's of course the stupid things that you can put in a [inaudible] record. That's always [inaudible].

Let's start with example one. Well, a lot of times we said that .nl and .se have a high level of DNSSEC deployment because of financial incentives. If you do DNSSEC, you pay less to the registries. Stuff like that.

The hypothesis of the research was that the incentives only benefit large DNS operators and not the small ones. So it will be encouraged – deployment on a big scale – but it won't necessarily

follow the best security practice. That's another part of … So I'll just do a tick mark, but that's what it is.

So, first we have to look at how the market looks in the Netherlands and Sweden. If you put that to the 80/20 rule, 80% of the signed domains are done by 14 operators. That's it. You got 20% by the rest of them. I think in nl there's about 1,200 to 1,500 operators. So that is really nice. In Sweden, just three operators responsible for 80% of the signed domains. I don't know how many operators are in Sweden, but it is, compared to what this is. It is just three for 80%, which is kind of … So there are different ways of how this is done in the market.

Okay. [inaudible] below and you can read it at leisure.

Okay. What is the proper way of doing DNSSEC? Well, there are a lot arguments about it, but there are guidelines from NIST that were mentioned in an earlier talk as well. That's being taken as the stick where you measure what is good and no, so to not into interpretation problems. So that's what it is.

The result of that is that the operators use way too small a ZSK according to the NIST, and they never roll for the ZSK. So a similar result for all large operators in Sweden and in the Netherlands. That thing actually shows you that. The ZSK is below 1024, and the rollover never happens. That's for the three top operators

from Sweden. That's one where you get more or less the same thing.

Okay. Now things are getting interesting. How do things go over time? I guess the Swedish registry decided to change the incentive policy and put requirements on how DNSSEC is being done. Actually, this was announced at the same time a talk like this was also given in Sweden. We saw immediately results.

So suddenly, that's what you see. There's a button for, I guess, light? Well, you see on the [inaudible] graphics the day the talk was given in Sweden suddenly – and the [inaudible] sent the results – you see the amount of DNSSEC going up for .com, .nl, and .se from one.com. That's a single provider. So they apparently listened to the talk and immediately took [inaudible] things do change.

The other thing we can see is on the right side. The green blob was what was there. That say – I think it was the same day – they actually removed DNSSEC from a lot of domains and switched the algorithm, and suddenly, instead of using the [1024] ZSK, they switched to the – I never can pronounce that one – the [inaudible]. I cannot read my own stuff. The ACDS [inaudible]. The curved [inaudible], which is way more efficient than the other [ISAkeys]. That's why [inaudible]. Apparently they did some work

doing that, and then suddenly they're back again with DNSSEC but with a new algorithm.

Another example is the DNS resilience. Can we measure DNS resilience? What you see here is the big data DNS attack on Dyn happened. That was in October 21, 2016. That's where you see suddenly some change. The blue line operates, which was using Dyn exclusively, and started to switch to a second operator or more. So the non-exclusively – you see also a difference there. It looks fairly [inaudible] but it's optical and about a couple of percent. But people switching to multiple operators were actually the big guys like Facebook, Twitter, and more of these, which had a big impact on what people call the Internet.

Anyway, so it's interesting to see that you can measure this stuff. There are more things about DNS resilience against DDoS attacks and a virus called MADDVIPR. This is especially done in cooperation with CAIDA and it makes a lot of use of OpenINTEL data to map points of failures and trying to figure how often they happen. Things like parent/child delegation mismatches, TLL mismatches, the shared infrastructure – how much people would [inaudible] sharing because use all the same profile – and topological bottlenecks in the network. So you see things like that.

If you look at the mismatched TTL/parent and child, you see that some people have quite a short TTL. So in a DDoS attack, they actually can switch providers, but they forget that, if the parent has a much longer period, it will be on the caches. Depending on the behavior on the resolver, this shortening of the TTL won't help against DDoS attacks and switching.

So DNS hijacks can also linger much longer. If the parent has changed and they won't go back to what it is supposed to do, well, there's this one- to two-day waiting time by a lot of big registries.

Another thing is the topological diversity. How many different [ASes] do we have? You see here the breakdown, the difference between .com and .nl. In .nl, half of the domains have at least two AS. In .com, it's 75%. So they are relatively more vulnerable. That's something you could say. This is always a bit of a handwave. I actually why .nl has so much but that's for something else.

The majority of .com and .nl has name servers and different prefixes, but only 50% in a single prefix. One of the projects just starting up is trying to figure out whether the name servers, even with a different prefix, are sharing the same location or not because a lot of people are sharing in the same data but having different supplies. Well, if the data center burns down, both of the

supplies go down. So things like that. Trying to figure out whether you can say anything about it.

Now we are coming from the TXT records. If you really want to do stuff and store stuff, you always put it in a TXT record. I know people who have put complete [IFCs] in TXT records. I've seen the [inaudible] lender is also published in TXT records. I've seen [inaudible] being welcomed in TXT records. They joined the club. And more of this interesting stuff. This is being done but is pretty harmless.

There are also things less harmless. You find HTML snippets. That might or not be on purpose. When I did some other research, I saw complete web pages that are very weird. You see also JavaScript, Window Powershell code, and other scripting languages, which actually can be done by accident or not. It also can take interesting scripts to do for botnets and other guys to talk to each other and automate things. There are a lot of ways of doing that.

Pem-encoded X.509 certificate, snippets of complete zone files. Something went out [inaudible]. You cannot make this stuff up.

[inaudible] looked at one of the root servers and looked at all the TXT records they got out of there. [inaudible] you find is very interesting.

But on the other hand, there's always Hanlon's Maxim: Never attribute to malice that can be adequately be explained by stupidity. Because people might do very weird stuff. That's life.

So here's the drum roll. What is the most stupid stuff people found according to OpenINTEL? Well, the winner is the RSA Private Key. If you read about the [inaudible]. And the question is, why? Why would somebody put a private key into TXT records in the DNS, which is seen by the rest of the world and is supposed to be private?

This actually is not something that happened at only one stage. We see this multiple times occurring on a regular basis. So what on earth are these people doing? How can this happens.

Well, we actually found it out as well. Ah, DKIM. Yeah. You have to put [a] key in there. "Didn't read the instructions that well." So note that we blocked out parts of the keys to protect the [DNS innocent], although somebody must have seen this before as well.

So this is what actually somebody of the whole project – I know I'm going a bit quick, but then people can have some break before the forum starts.

Anyway, so really need to boost the data archival on the long term so we can find long-term [inaudible]. What actually people really

want to do is expand this also to ccTLDs. That's because now all the data comes from the CZDS stuff from ICANN, which is a problem in itself. But let's not go into that now.

The long-term goal is trying to become the long-term memory of the DNS. So if someone in 2025 wants to know what DNS looked like in 2015 before, they actually do see enough data that they might find an answer.

The other thing is trying to improve the performance and the security of the DNS by looking at what people are actually doing. One the things actually pop up – for instance, people say, "Well, emojis in DNS. What harm can it do?" That's one of the things. So another student project just started, trying to figure out how these things are used in practice, which is actually not a trivial affair. You constantly have to look at a lot of things [used to] phase them out of the [inaudible], the other stuff.

But that's about the end of this. Are there any questions? If you want any more information, go this webpage.

Questions.

EBERHARD LISSE: Thank you very much for explaining that not everything is malice.

WES HARDAKER: Wes Hardaker, USC ISI. Everything is malice in my opinion, but onward. Thanks for doing this. It looks fantastic. A couple of things. I guess most importantly, can you explain the difference between OpenINTEL and the active DNS project, which you didn't talk about, from Georgia Tech, where they're doing something similar? They're trying to measure all domains they've ever seen and have been doing it for a number of years.

JAAP AKKERHUIS: I don't really know where there's any contact between these. They're working with CAIDA. So I really don't know. It's difficult to answer – ah. Here's a reaction from Roland. He says, "Well, we started before Georgia Tech started."

WES HARDAKER: Fair enough. Hi, Roland.

JAAP AKKERHUIS: But he's willing to talk to you offline. So that's what I just get here.

EBERHARD LISSE: Patrik?

PATRIK FALTSTROM: You talked in the beginning about the high number of signed zones in Sweden. What we are proud over is much, much more the high percentage of validation, not signing. We do know we have a lot more work to do regarding signing zones.

JAAP AKKERHUIS: Okay. Good point, because … People, if they don't [inaudible] that's a good point. I've got a [tea leaf] for the KSK after it got retracted. Why is it so high still? That's a question for the [DS]. It has to do with that. In our countries, it's because registries are totally separate from the infrastructure and resolver providers. It's a totally separate thing. When are fully from the top-down [inaudible] compliant, then we will talk to the regulator. They're keen on regulating every single resolver provider at the same time because, if you don't, then the ones who don't do it have got a commercial advantage.

Signing is one thing. Resolution or validation is done by somebody totally else. So that's a different approach that one has to take, I think.

Any, the comment from Roland is that OpenINTEL is helping IIS to improve their incentive.

EBERHARD LISSE: Okey-doke. Thank you very much. Now we have the presentation from Photochanan Ratanajaipan. I can't read that. I'm so sorry. I apologize. To be honest, it happens in my own country. In my wife's language, names have meanings and many consonants and I'm unable to address my patience with those.

PHOTOCHANAN RATANAJAIPAN: Good afternoon, everyone. We are from Thailand, THNIC. This is [Ms. Pense] and me, Photochanan. Thanks to ICANN for letting us share our view of EAI and IDN in this session. Thanks, John – maybe not here now – the presenter of the session before this session. He explained everything on how to make your mail EAI that made our presentation much easier because we will share our view in a little bit of an [overview] [inaudible] of experience in implementing IDN and EAI in Thailand.

In Thailand, we have a problem of the digital divide. Some part comes from our English literacy. We approximate that we have an English proficiency that's over 20%. We believe that using Thai e-mail addresses and IDN could improve our Internet presentation and also facilitate communication between our Thai people and government agencies to Thai people also. That is our view.

Today I will update on the status of our Thai IDN and Thai EAI. Apart from facilitating domain name registration in THNIC, our

organization, we manage a ccTLD and we also support study, research, and development on the Internet in Thailand. We also enhance of Internet technology to Thai people and, of course, we include the IDN and EAI and universal acceptance knowledge to Thai people.

As some of you may know already, ICANN defines universal acceptance, or UA, as a state where all [valued] domain names and e-mail addresses are accepted, validated, stored, processes, and displayed correctly and consistently by our Internet-enabled applications, devices, and systems.

This is our [profile].

Okay. Let me start with our timeline that we implemented. Since 2011, our IDN ccTLD string [that's called Thai in Thai] script, was successfully evaluated. Now we have about 18,000 IDN domain names.

We launched Thai EAI in 2016 and have provided Thai EAI .accounts for around two years. We work with a Thai company called [Throughwave] to provide an EAI e-mail system that's fully developed to support sending and receiving both English e-mail addresses and also Thai e-mail addresses and, of course, EAI e-mail [or so], not only for Thai language.

In 2017, we started our project called [Kunda] Thai. It's a project to provide free e-mail accounts and mailboxes for Thais.

This is the number of domains. As you can see, it's not the million that other people are talking about. We have the number of IDNs domains. Compared to the ASCII base domains, it's about 75%. For IDNs, 27% are ASCII-based domains. A proportion of domain categories in each categories of the seven categories that we have is with the same proportion. A big part is from [c-or]. That is equivalent to .com, but in our country, you use two [inaudible] co. The Thai domain name is, in other words, is comparing to each other.

We have e-mail accounts on a UA-ready e-mail system, about 3,000 accounts. More of than half of these Thai e-mail accounts.

THNIC itself has tried to promote using IDN and EAI by providing free IDN Thai domains for new domain registration. That means anyone who comes to register for an English domain name, we provide a .thai domain for free. So it's like a pair of that one.

We also have a program for a free IDN .thai website and a booklet for unseen tourist destinations. So we go [somewhere in Thailand in which] we have unseen destinations. So we give them the name in Thai and prepare a booklet for [splitting] out into any organization in timeline. That is a kind of promotion activity.

We provide free Thai EAI accounts for pilot groups and new domain registrations as well. So that is how we promote it.

About awareness-raising activities, we have been working to raise awareness of the Thai top-level domain and EAI and to educate the technical community on the issues of UA. Many awareness-raising activities we have done with our partners, which consists of local software developers, government, CIOs – maybe not much nowadays because we are going to have elections soon. In many countries, they may find the same problem. When we have elections, they don't have to talk to us anymore. They would like to [us] change the future first. And other local Internet communities.

Other than the normal PR activities, like using website, social media, and exhibition booths, we also talk to the Minister of Digital Economy, which is a ministry in Thailand, and the Department of Business Development, which I think may be a good promote new business registration or for even the old business to have the new Thai domain name also.

We organize seminars and workshops for public and technical groups, such as developers which support successful implementors. They were invited to share their experience. Sometimes we invite from success full implementors from India and also China. We also have a coalition with them.

We believe that interesting and useful information from a group from ICANN is already well-documented. So we try to make it into Thai, translate it into Thai, so we don't have to invent new content but just translate into Thai because UASG has a good system for documenting those things.

We have coordinated with the USG to translate the Quick Guide and also, more and more in the future, I know that UA is doing more on this.

Some group of organization names were also reserved. So are aware of maybe conflict if you don't reserve some name for them in Thai, like government agencies, schools, or hospitals. So we reserve the names if they are ready for getting the Thai name. They just contact us and then we can release that name for them.

Just this last December, we also had a camp that was called Thai Network Group (THNG) that we organize annually. But last year, we organized it in the theme of Internet governance and hack-a-thon. In the hack-a-thon event, we let them understand and know what the problems are of an e-mail system that is not EAI-ready and then let them try to config the e-mail server and e-mail transfer agent and also test sending and receiving from and to Thai e-mail addresses with many configurations until they were confident that they could use that knowledge from the camp for

their real job in the future, for system administrators of those things.

This is our direction. We aim to convince that that e-mail address, especially the Thai e-mail address, can be used a digital ID for Thai citizens and also for corporate, and, in the future, maybe other things in the IoT world. Together with an English e-mail address as a global digital ID [when they come].

Okay. So that is our direction and we will keep doing that to the – sorry. Before doing that, we have to make sure and evaluate ourselves that all our system that we have are already UA-ready, or Universal Acceptance ready. So we have to meet that. We are not in that state now. We are improving that better and better.

We have a plan to have another Hack-a-thon, but now it's not for teenagers or the students groups, but we will expand it to the real developers in Thailand to come together and update related APIs and library tools to be UA-ready because we have done a survey from them before that. They normally use those things. They didn't invent every single code by themselves, so we will find the common API and library that they normally use so we solve the problem from that root. When they use that one, they don't have repeat doing those things again to be UA- and EAI-ready.

This is just an update by picture of what we have done about seminars and workshops and promotion booths for the e-mail

Thai. In every booth presentation, we normally distribute a Thai e-mail address for participants.

This is the booklet that I mentioned that is in Thai but the story inside is about the tourist – the information for them. So it'll be easy to distribute to everyone [to]make interest of it.

On the right-hand side is the website that [inaudible] with the booklet. We have done that for almost ten projects already. This is the teenager camp that I talked about, the THNG. It's [related] to Internet governance and localization, cyber bullying, GDPR, and also technical knowledge about net worth and making things to be EAI- and UA-ready.

We have to thanks ICANN again for support financially and also for our staff to join the event and give knowledge to the event.

Okay. That is the end of our slides. You can refer to our use case documented at UASD013A. That is the case study of developing Thai EAI that UA [recorded].

We will keep supporting the Thai community to acknowledge the EAI and IDN and [inaudible] to implement Thai EAI- and IDN-ready software. That may be, you have to understand, not the same label as fully UA-ready because you know that it may be a problem of many scripts that we have to handle about to be fully UA-ready, like left-to-right, right-to-left scripts. Something like

that. But we pay attention onto Thai EAI first to make sure that, in our country, we can use that.

But we are a small group of users and it depends on [inaudible] and service. We still depend on using them things from [inaudible]. So I think ICANN could support and engage in any [inaudible] in that body and also [inaudible] so it's provided globally to make things to be UA-ready so we don't have to put much effort on that.

So thank you very much for your time. You can share with me.

EBERHARD LISSE: Again, I'm very [inaudible] [from the chair a little bit]. I'm not a great fan of outreach because it usually doesn't work. Does it work? Your outreach with the camps. Do you see improvements afterwards.

PHOTOCHANAN RATANAJAIPAN: We haven't measured the result of success of outreach in detail, but for some projects, it worked. For the booklet, I think many Thai [districts], in other regions of Thailand, our project proposal was much better. So there was a lot of application for doing those Thai domain registrations more and more.

EBERHARD LISSE: Okay.

BARRY LEIBA: Hi. I'm Barry Leiba. I can make these arguments, but I want to hear you explain why it's important to you. So a two-part question. To what extent, as people can send e-mail with Thai in the subject and Thai in the text, is it so important to have it in their e-mail address?

The second part of the question is, to what extent do they need to be able to communicate non-Thai e-mail using a Thai e-mail address? Or is it okay to say, if you want to communicate with non-Thai people, you need to use your ASCII address.

PHOTOCHANAN RATANAJAIPAN: So the reason I mentioned on the first slide the real [inaudible] in Thailand because of the English proficiency is quite low. We [inaudible]. Many people don't speak English. So we hope that they are having domain names and also e-mail addresses in Thai. It could improve communication, I think, for the domain name itself first. When we like to contact government agencies, normally they promote their [service] via the printout or by radio. "If you don't know this Thai, please

contact blah, blah, blah." But we have to tell everything in English spelling.

So I think communication in other ways not using the Internet – printing things or other media. Explaining in the Thai language is much better to understand for Thais.

EBERHARD LISSE: This question – let me finish – has been asked at the Beijing meeting, where we had a similar presentation. The presenter said that she was unable to communicate with her mother in the rural areas because she doesn't speak English and they can't use e-mail My mother-in-law in my area can hardly read and she speaks no English. But on her cellphone she has figured out to do this. But that's an actual advantage that the people who can at last speak Thai who don't know the Roman alphabet easily have. They can easily communicate with each other.

BARRY LEIBA: Than you. I wanted to hear it from them directly.

PHOTOCHANAN RATANAJAIPAN: Okay.

BARRY LEIBA: But how about the second part of the question. Do they need to communicate with non-Thai people using a Thai e-mail address? That speaks to the interoperability issue.

PHOTOCHANAN RATANAJAIPAN: Actually there's no need. We create both Thai e-mail addresses and English e-mail addresses as alias. So the system could – they themselves, if they don't know anything about English, they don't have to use English for communication in Thai. But if they prefer some [inaudible] officer, they can use English. So they can select in the header that they would like to send mail out in an English e-mail address.

EBERHARD LISSE: To be really honest, I do not understand your question. If you don't speak – let me finish – English—

BARRY LEIBA: I'm letting you finish [inaudible]. Go ahead.

EBERHARD LISSE: If you don't speak English, then how can you communicate with somebody who doesn't speak Thai? That's what I'm trying to say. The question is, if somebody sends me an e-mail from a Thai address and a hit the reply button, the question whether that

works is valid. But the question whether somebody can communicate with an English speaker is not valid, as far as [inaudible].

BARRY LEIBA: The last part is exactly what I was getting at. If they don't need to communicate with non-Thai people with the Thai e-mail address, we don't have the interoperability problem that we would otherwise.

So part of the answer was, for those who know English, they have an English address to use. That's the answer I needed to get aired.

PHOTOCHANAN RATANAJAIPAN: Okay.

BARRY LEIBA: Again, as I said, I could make these arguments. I wanted to hear it from someone who actually had the issue.

EBERHARD LISSE: Thank you very much. That leaves us to give the latest.

PHOTOCHANAN RATANAJAIPAN: Thank you.

EBERHARD LISSE: Warren with give us a resume of the 38th Tech Day. I have figured it out now. It's the 38th. At least I've got 38th agendas from my different agendas on my computer.

WARREN KUMARI: This is going to be a short summary of many of the presentations for people who weren't able to see them all.

The first was a really interesting IPv6 presentation by Microsoft, who is making some really good progress on their IPv6 deployment for a number of reasons, partly for [dog fooding] but also because stuff like running dual stack is expensive and complex.

They had some really good discussions on why they're doing this, and they started off by rolling out IPv6 on their wireless network and allowing users to opt in. The goal is to [be] IPv6-only everywhere. Hopefully, they'll get people to opt in more and then eventually force of their employees into using it.

The next presentation I also thought was really good had some fascinating things which I didn't know. That was two million registered IDNs, which is one percent of contracted names. That was something I had no idea would actually reach that sort of

level of IDN deployment. Sadly, the distribution is fairly uneven, but still.

What I thought was interesting is that there are around 5,000 names which are invalid IDN2008. Most of them are length errors, which isn't something that I really realized. But what's said is there are still some sort of naughty gTLDs allowing name, some of which are sloppy, but also sadly, some of things which clearly things like phishing, etc.

Not sure if Paul's in the room. Paul Hoffman did a fascinating presentation on how to find a DoH resolver. Sadly, the DoH standard doesn't really explain how you should do this. And Firefox in conjunction with Cloudflare provides a way to easily enable DoH, but they've defaulted to providing Cloudflare as their DoH provider, and it seems tricky to change it.

Eventually, there may be an easy way to get added to a list of available DoH providers, but it's still not clear if that's actually the right answer, if there should be a list or if people should just [do] it some other way.

Paul has a proposal which is in the IETF on how you'll find a DoH server associated with the resolver.

There was what I can call a fairly healthy debate about a lot of this.

Next was the public suffix list discussion. On a sort of personal note, thanks very much to the Mozilla community for actually running and maintaining and keeping the public suffix list going. This is a really critical service for things like cookies, [lits and crypt] etc., one of those foundational parts of the Internet that everybody kind of forgets about until they need to interact with it, at which time they really it's a really important and useful resource.

In that presentation, there was a plea for people to please help maintain it, come along and contribute, volunteer and help run this.

The host presentation was another one which I thought was great and fascinating. JPRS is doing a lot for being able to actually survive disasters, and also they've got one of the more interesting disaster recovery plans I've seen. I've reviewed a lot of those, but things that [I] almost always seem to miss is things like, if there's disaster, do you know how to actually get to the disaster recovery site? Can you do this without walking under bridges which might not actually be standing, etc.? That I thought was a great thing.

Also, a lot of interesting stuff using the .jprs gTLD to try and do research, and some of the difficulty of doing that with the ICANN contracts and what you may or may not do with them.

There's also the Tim April SSAC emerging security issues talk, which has lots of good advice on how a registrant should keep their domain safe and includes a lot of good advice for registries and registrars.

Unfortunately, as Tim says, there are no silver bullets. This requires a defense in depth thing. But it also had a lot of good stuff, not just on how to protect yourself but also how to monitor for issues, including things like [inaudible], etc. And there will be a much longer talk on this on Wednesday, I believe, in the afternoon.

Another one was how to make your e-mail EAI-ready. There are actually two somewhat related talks on this, both this and the IDN and EAI in Thailand talk. This is had a history on the move of e-mail from non-ASCII to be able to actually support Unicode e-mail. And also some of the weird and interesting compatibility issues with legacy e-mail [EAI]-capable systems and what you need to do to make yourself compliant for EAI stuff.

Probably my favorite talk was Jaap's long-term DNS research talk. I really enjoyed that. Passive DNS, which is a useful and very powerful tool, only sees what clients ask. There are long-term research projects actively probing [260] million domains per day and sucks in zone files to find them. I really enjoyed this and the plea to contribute.

The IDN and EAI in Thailand talk was also really interesting, both how quickly deployed. 27% of TH domains have a matching IDN name, and it seems that much of this is through the advocacy and pushing from THNIC. And also it was interesting that they're moving towards trying to use e-mail addresses as a global digital idea.

Also, on a personal note, the last IETF was in Bangkok. If you've never been, it's a wonderful and fascinating city. I really enjoyed my time there.

As the final wrap-up, this has been an interesting Tech Day covering a wide range of things. Thanks to the speakers for coming along and being well to present, and also to Eberhard and everybody who helped organize and corral and do all of the agenda management and everything else. Thank you.

EBERHARD LISSE: Thank you. With that, we're done.

[END OF TRANSCRIPTION]