BUENOS AIRES – How It Works: Internet Networking
Sunday, June 21, 2015 – 13:00 to 15:00
ICANN – Buenos Aires, Argentina

STEVE CONTE: Welcome to the third and final "How It Works" series, the pilot program that we're running on talking about the protocols underneath the technologies that we're talking about all this week and in general at ICANN.

David Conrad isn't able to make this one, so we're stuck with doing self-introductions. We're both at ICANN. I'm Steve Conte. I currently work with the Security, Stability and Resiliency Department, but I've been in and out of ICANN for quite a long time.

I used to run a root server with another team at ICANN. I've been head of IT. I've been head of security, so kind of around the block a couple times. Right now, I'm currently working on training and outreach and stuff like that.

I want to pass it over to Ed for an introduction here, too.

ED LEWIS: My name's Ed Lewis. I've worked for ICANN staff now for a little while. My background in this area started out with DNSSEC. We'll talk about one of the protocols where I was one of the original developers of the code to prototype that for the standards work that went on in the IETF.

Also, I was one of the working group chairs. There were two co-chairs for the EPP protocol, which you'll hear about here, also. Operationally, I've worked for RIRs. I worked for ARIN for about two-and-half years, I guess, a couple of years ago. For the last ten years, I've worked as a gTLD and ccTLD operator. So I've seen all these protocols used in many different ways, and I've done talks on this going back quite a while.

STEVE CONTE: Yeah. Have a seat. I see many people here who've been to all three or the other two sessions already. That's good. We're going to be building off of those. Russ gave a great session about the IETF and working groups and how the process is and how the protocols are made. You'll see a lot of references to the IETF throughout our presentation.

Alain mentioned talking about different network layers of the OSI model. You'll see that at ICANN we're talking about what Alain said was the layer 9, the political layer. There's certainly a political or policy aspect to each protocol that we're going to be talking about, especially within the registry services.

We're not policy makers. I have no intention to be one. We'll talk about WHOIS, for instance. We're not going to talk about the policy ramifications of WHOIS. I do have a slide that will show you where those sessions are. I encourage you, if you have interest in the policy aspect of any of these protocols, that you go and participate in the policy working groups. We're strictly going to talk about the tech, and

hopefully that's exciting enough and we don't have to go into policy discussions.

What is a domain name registry? Specifically, it's a single location that you can associate domain names to a specific top-level domain. It's basically like any other registry, just a single location where you can list and provide a list of those names that are underneath that registry.

Within the tech space, TLDs, top-level domain spaces, are also called a zone. When you're producing information for the computer, you produce a zone file for that DNS server, and it uses that to communicate with the other DNS servers.

Obviously there's a whole bunch of registries. As Russ mentioned this morning, there's the IANA. That's the registry itself. That houses unique protocol parameters. There's the regional Internet registries. Alain spoke about that. They house networking addressing and routing information and autonomous system numbers. There's protocol parameter registries, such as IANA. We just spoke about that.

If you own land, you have to register that within some governmental aspect. If you own a vehicle, you have to go their Department of Motor Vehicles to get their license or to register their vehicles. And you know there are gift registries and wedding and baby registries, all kinds of different registries.

We'll specifically be talking about what takes place within a TLD registry today.

If we look at it in comparison to what a registry is inside the hierarchal DNS tree, we see at the very top is the root zone. That's the 13 instances or 13 letters of the root server, of which there are 140-something, maybe more, instances of those root servers out there now. Those only handle points downward one level on that tree, and they point to the registries. That's the top-level domains.

The registries house all the registration information from the registrar and the registrant. The registrar is going to be the middle man between the final customer and the registry. So if I wanted domain. tld, I would be the registrant, first of all. I would register that through a registrar, who would then be the discussion point and the communication point between me and the registry.

This is kind of just what I spoke about here. As a registrant, I have two ways to get into a registry, for the most part. In some cases, the registry can be a registrar. They can serve their own and sell their own domain names. For the most part, though, on the Internet, the registries will go through registrars. They'll have contracts with registrars to resell names though that.

As someone who wants to register a domain as a registrant, I have basically two choices. I can either go to a registrar directly, such as Easy Domains or GoDaddy or Joker or whatever's out there these days. I can register the names through them, and then they would talk to the TLD registry, or I can go to a reseller. A lot of registrars – I believe something like Tucows and other ones – have reseller agreements. As

a value-add, there would be reseller organizations that I could then go buy a domain from them.

A lot of web hosting companies are also resellers, so that's the whole one-stop shop aspect where you can go and get your domain. You can house it and build your website and have your e-mail, all from one location.

I'm going to go back one slide. This is from a customer perspective. But with the other side of the registry, also we talk about data escrow. What happens if a registry has a business failure or some other kind of failure? There has to be some kind of data escrow that takes place.

There are hooks into a registry from data escrow, and then there are also trademark clearinghouse aspects of having a registry, too. Ed will be talking about both of those in greater detail.

Here's a quick look at the protocols that we'll be talking about today and how they relate to the TLD registry. We're going to start with DNS. Does anyone here not understand how the DNS system works? It's okay to raise your hand. We can go over it really quick. Everyone is a DNS expert? You'd like an explanation on DNS?

In the last session, Alain was saying routing is who you know. It's a referral system. DNS is very much like that, too. If you want to get somewhere, you ask the top of the tree. If I want to get to Ed.com, I'll go ask a root server and say, "Hey, how do I get to Ed.com?" and the root server is going to say, "I don't know. No idea. But I know how to get to .com. Go ask them."

So I will go to the registry, the .com, and say, "Hey, I want to go to www.Ed.com," and they'll say, "I don't know, but go ask the name servers for Ed.com. Here's the address."

Then I'll finally go to the name servers for Ed.com and say, "Hey, I want to go to www.(which is a host).Ed.com." Ed.com's name servers are going to say, "Oh, I know that answer. Here's the IP."

So I'll take that and then I'll go surf the website that Ed's got up and buy whatever shoes or anything else that he's selling on it and do that.

I'll cache that information because chances are I'm going through an ISP. The ISP is going to be my stub resolver, and it's going to cache all the answers to all the questions that its customers make. That way, in case somebody else inside the ISP wants to go to Ed.com as well, it doesn't have to do that entire query response again. It'll go to the stub resolver and it'll give them the answer.

Does that make sense? We just talked about this. A DNS protocol is a look-up, kind of like looking up someone's name in the phone book to get their phone number. The query asks for information: the domain, the type. If I'm sending an e-mail, I'm looking for an MX record versus if I'm looking for a host (www), I'll be looking for an A or Quad A record.

It's going to give you a response, and that response is either going to be positive, which will provide you with information, or it will be: "No. There's no such domain, and don't keep asking me that question." My son gives me a lot of negative responses like that.

DNS is one of the earliest protocols, so it has impact on almost every other protocol or any other application that's out there on the Internet. Because of that, it makes it kind of a critical infrastructure because humans are the ones who use the Internet. Well, machines do the work, but the humans who want to use it can't remember IP addresses. Especially now that we've moved to IP version 6, it's that much harder to remember an IP address.

If I'm trying to go to ED.com by IP address and he changes hosts and gets a different IP address, I'm going to be going to the wrong machine. DNS makes that easy because it allows us to map a name that we commonly use and not have to worry about the number and the address behind it.

Registries exist because of this because there are maybe hundreds of registries out there, TLDs? Over 900 registries out there. It's because of the DNS that this opportunity has taken place. It's because of this that we have the ability to map names and numbers.

Also, as I mentioned with the example of Ed changing providers, it gives us a sense of resiliency, too. We know that as long as there's a mapping out there and that mapping is accurate, we can continue to go to the service that we're looking to go to.

Here's just a simple diagram talking about the players within DNS of the registry. The blue that we see is the registry itself. The orange that you'll see – this will be the case for all of our diagrams – are external parties.

You'll see that the registrar, which again is the agent for the registrant – if I'm registering a domain name, I will go through a registrar, and he'll act on my behalf to go talk to the registry. I will register through a registrar. The registrar has a communication with the registry to register that domain name and put in the relevant information.

Then that's put in, and then down in the yellow is where protocol magic happens. Once it's in the registry database, that's where DNS servers will talk to one another to get the information out.

Then over here with no arrows we see the IANA. The IANA's involved with registries and with DNS because they house the unique identifiers on the Internet. They also are the ones who manage the root zone. There are a couple other things as well that they do, so IANA hovers around the registry process I guess is a good way to say it.

The components of DNS: you'll see these three definitions talked about a lot throughout the week and throughout the ICANN ecosystem. The authoritative server is the server that a registry operates. It's actually the server that any level operates. If I'm running Conte.net, I'll have my authority servers, authoritative servers, that are authoritatively talking about any host that's within my domain.

If we bring that up a level, .net is the registry, the TLD registry. It has its own list of authoritative domain name servers that are talking about the data within that registry, within that zone file, with authority.

If we move that up another level, we talk about IANA and the root. The root is an authoritative server, and it houses all the name server

information for the registries only. We can keep going up that tree. Each one of those is an authoritative server.

A recursive server is what you'll see in an ISP or the famous 8.8.8.8 from Google if you want an open DNS resolver. The recursive server is mostly the servers that will ask questions on your behalf.

When you turn on your machine and you go to a website, your machine is pointing to some other set of DNS servers, and those are the recursive servers. They will ask the question on your behalf and then come back and give you the answer.

Those are also the servers that are going to cache the information because chances are there's more than one customer that's using that, and they want to be able to serve answers very quickly and in a reasonable amount of time.

Then there's the stub servers – the clients – and those are the machines or whatever processes at the end of it. That also has caching, but most of the caching that takes class is at the recursive server.

When I get the address for www.Ed.com, I'm going to keep it too for a certain amount of time because if I ever go back again, it just makes the query answer tree smaller because I'll know the answer for a certain amount of time.

Here we have a diagram of the different servers in relation and, again, yellow is where protocol magic happens. If we just take this to the same map of looking up a domain name or a host, www.MyHost.com

or whatever, you'll start moving upwards. You start at the top, which is the authoritative servers, and there's going to be some circling within the authoritative servers. In this case, it's blue because we're talking about the registry only.

Then down we go, that's going to be outside of the registry, but that's where the recursive is, the ISP, and the stub client, which is the machine. All of that takes place with DNS in the middle.

DNSSEC, Ed, I'm going to turn it over to you.

ED LEWIS: Great. Go right. Go to the right. Alright, so you just had an introduction to DNS. DNSSEC is an extension on top of DNS. It's not an entirely different protocol. It takes the DNS protocol and the way that it behaves, as Steve had mentioned, and has addressed some of the problems we had that we foresee with the security of it. It doesn't address everything, but it addresses some of the problems we saw there.

To start with, what does DNSSEC do? The first observation is that, as Steve was just describing, rarely does the end user of information for the DNS – anyone who does a look-up in DNS for something – do they ever actually go to the source. They don't go to the one place that has the information. They go somewhere else. The DNS is built upon having these middle boxes throughout the system.

Because of that, we can't use what we use in the web and in a lot of other client server systems, where we just have the server secure the

line to the clients, saying, “Here’s your information.” The DNSSEC has to go and build onto its answers information so that as it goes through the system it, it doesn’t get changed.

Where does this come from? Right around 1990, it was noticed that because of the way the DNS interacts with intermediate systems, that was a big vulnerability in the system that you, as the end customer, the client of all this, want to find information and you go ask somebody in the middle, they may not go to the right place for the information. Or they’re very gullible. They could believe information from any place out there.

For the next five or six years, there was some basic research done, and then it went into more operational research. By 2004, a little over ten years ago, there were finally specs that came out saying, “Here’s how we’re going to approach DNSSEC.”

2008 was a big year for the history of DNSSEC because that’s the year that someone gave a talk, Dan Kaminsky gave a talk, illustrating why leaving the DNS unprotected, having DNSSEC on the shelf but having DNS out there unprotected, still left us vulnerable. He showed so many ways you could attack these middle boxes. It brought this question back into the forefront that we had to get DNSSEC done.

In 2009, they started appearing in TLDs and the root zone was signed in 2010. The approach, the way the problem is solved in DNSSEC – the problem being, “How can someone verify they got the right answer from the right source?” – is to use digital signatures.

What happens is we take the basic DNS response, which is unchanged, and we add to that a signature. A digital signature is kind of like a cryptographically-based checksum that if it doesn't validate, you know you have the wrong answer. You don't know what the real answer is supposed to be if it doesn't validate, but you know you have a wrong answer.

Behind all this, I need to have a public key. I have a digital signature over around my data and the public key that validates that. Public key cryptography is a way of signing things where I can give you a public key. I have a private key, I sign with a private key, give you the public key, and then you can validate that I actually signed this.

If the public key works, it means I have the private key. You can tell it came from me. Now, that's just the first step of what DNSSEC is doing.

What DNSSEC does is it takes that and builds it into a hierarchy by having a piece of data signed by public key that you get. Well, is the public key valid? Well, the public key itself is signed by another public key and so on up the tree, mirroring how the DNS is put together.

To a registry, what does this mean? The registry's first job when you get into cryptography is managing cryptographic keys. That's not an easy job. You have to create the keys. You have to use the keys for some time. You have to put them away. You have to make sure you're managing all this. You have to make sure the private key stays private. That's kind of important here. If the assumption is that the private key is only you know it, it better stay that way. A lot of times that's where

the [inaudible] that that private key becomes public, and then it's worthless.

The next thing a registry has to do is it has to be able to help build the hierarchy of trust with its registrants. The way this is done in DNSSEC is a thing called a DS record. The DS record doesn't really matter – the detail of it – right now, but the idea is that the registry has to be able to get the DS record from the registrants into its system, just like it has the NS records, which helps to point up and down the tree.

A TLD hosts NS records for a lower layer by saying, "Go there for information." The DS records saying, "Go there with the security information." They kind of run in parallel.

The third thing that the TLD has to be able to do is it has to be able to sign these DS records. It has to put its digital signature on that with the private key it has and publish this all out on the DNS systems it has out there.

Also, I'll mention the "no" answers. No answers are something that we don't normally think about when we describe a system and a service, but in DNS, a lot of times people look things up that don't exist. They say, "No. They don't exist." Saying no has to be signed also, and the problem in DNS was the DNS used to have way of answering no by saying nothing, and you can't sign nothing. You have to have something to sign, so they had to create a whole way of saying no and putting signatures on top of that. That has been as much work as anything else.

Finally, the registry has to interact with IANA more often than it used to to update the TLD's own DS record for building this chain. Initially, IANA only [needed to] know the NS records for a TLD, and if that never changed, there may be no other interaction. Now with the keys changed, we have more interaction with IANA.

This picture is close to the previous picture of DNS, showing the orange bubbles being external to the TLDs, the registrars out there who are putting in DS records to the DNSSEC functions area. IANA registry is where the information about where the DS records go up the chain. The functions put these into the database. The database [fills] the DNS server, and now DNS server is speaking DNSSEC. Actually, it speaks DNS and DNSSEC at the same time.

Before I give it back to you, to kind of put DNSSEC and DNS together, my favorite analogy is that a DNS is like a car and DNSSEC is like seatbelts. It's just a small piece of security for when you're driving a car around. Though the seatbelts in this case are very complicated and a lot of work, they save you when there are problems out there. So DNS and DNSSEC work together like seatbelts in a car.

UNIDENTIFIED MALE: Before you hand it over, can you talk about whether DNSSEC does encryption? Is it security in the fact that it's going to protect the data during a transfer to make sure that only you can see the data?

ED LEWIS: It does not do encryption. It uses cryptography. The difference is that the way it's using cryptography is providing some string of bits that, if you take the string of bits, the piece of data you have, and the public key put to some mathematical function, the answer is zero. Then the public key says, "Yes, I'm backing that piece of data."

The piece of data is always visible to somebody. It's basically a checksum or check bits on that data, but it does not hide the data in any way. It doesn't who's accessed the data. It doesn't hide what they're getting. It doesn't hide a lot of things. It's all out there in plain text.

The cryptography is only so that I can make sure that the originator signed the statement saying, "This really is my voucher. This really is what it's supposed to be."

STEVE CONTE: WHOIS. How many of you here have registered a domain name? Okay. Three people. Four, five people. We're at an ICANN meeting. How many of you have ever gotten e-mails (who have registered a domain) from the registrar or the registry saying, "Hey, you have to make sure that the information in your WHOIS data is correct because it's in our contract to make sure that the data is accurate"? Okay. Five people. This is good. We're on a roll here.

Back in the old days, when the sun was shining and birds were singing and nothing ever bad happened on the Internet, this actually worked pretty well because if there was a problem with a network or with a

domain, WHOIS was created to give one operator an opportunity to go and query who they should talk to to have a conversation to rectify any kind of situation.

It worked great back then because, again, the sun was shining, the birds were singing, nothing bad ever happened, and no one ever scraped records and used that e-mail for spam or anything like that.

WHOIS was created in a very simple form because it was meant to serve a very simple purpose, and that was to help troubleshoot and do other things.

It was around even before DNS. Ed, I'm not sure about that point, so I don't know if you want to speak to that specifically.

ED LEWIS: It was around before DNS in a sense that before DNS there were other forms of mapping names and numbers. At that point, there still had to be a way to look up who had what.

STEVE CONTE: Thanks. As I said, it was a very simplistic question and answer process. "Hey. Who's the admin contact for Ed.com?" "Oh, it's that." "Okay, I'm going to call him up or send him an e-mail and say, 'Hey, I can't get to your website,' or, 'It's down,' or, 'E-mail is not working," or something like that.

At that time, there were no concerns about privacy or security or accuracy because everyone was playing nice and the amount of

players was relatively small. It was still in this little unconceived, pristine, baby stage – aren't they all cute? – and then it grew up. WHOIS is still in its simple form, and the needs now are different.

But what does it do overall? It opens up a TCP connection. Remember we spoke with Alain last session about what TCP is. It's session-based. It's a hand shake protocol compared to UDP, which is almost like a fire and forget protocol.

You open up a connection on Port 43 and you ask it a question and you wait for probably years or twelve milliseconds, whichever comes first. It gives you an answer and it closes the connection and the application that you used parses that answer and prints it into a nice pretty format which a human can read.

That's it. That's what the protocol does. It was meant to be very small. It was meant to be very light because it wasn't passing a whole lot of data.

Now, from a registry perspective, it has some form of a WHOIS database within that registry. Now, there's two common forms of WHOIS that are out there. There's something called Thick WHOIS and Thin WHOIS.

Thick WHOIS is the traditional, legacy-style of WHOIS where a registry will house all the registrant WHOIS data within the registry itself. It has a very large – hopefully, if it's a successful TLD – WHOIS database that, when you go to ask a WHOIS question, it will give you that answer from the registry.

Now, a Thin WHOIS model is where the registry refers the query on to the registrar on record who the registrant has chosen. If I register my Conte.net and someone goes and does a WHOIS for Conte.net, the .net servers probably are pushing it off to the registrars, saying, "Hey. Go answer this question." It refers my question over to another server, and then I have to go ask that question to another server.

This works really well because, in some ways, the accuracy theoretically is improved because the registrar has the direct relationship with the registrant, with the customer, whereas at a Thick WHOIS level, there is a gap between the registration, the WHOIS database, and the registrant, the customer. So the accuracy theoretically is better.

The query time arguably is a little bit slower because of the Thin, because of the referral, so you have to go ask the question and then you have to go ask the question again. It's arguable how fast or how slow that's making it.

So there are different models, different TLDs. Different registries use these models in different ways. I think it's their choice. Is that true, Karen? No? Okay. I'm not going say anything on that then.

Then here in the diagram, we see the WHOIS client. If we go back to the question and the protocol, the WHOIS client is the external party. They ask the WHOIS server using protocol magic, and they're going to get the answer from some WHOIS server, via at the registry level or at the registrar level.

We find that free form and these kinds of questions, again, it's a legacy protocol. It's using ASCII only. We're now in world where we're talking about internationalized domain names and non-ASCII scripts on the Internet. It's becoming harder to address those kinds of questions in an ASCII-only world.

Differentiated access impossible. That's more of an access level of if I'm Steve Conte and I want to know about the details of a domain but I'm just an individual user, the registrar or the registry probably wants to send me a different subset of information – not as personal, not as private – than if I were some authenticated user for public safety or law enforcement or another aspect, where they might have more information that they could give to that user.

I'm hesitating here because we're getting into the policy talk and this isn't a policy discussion. But these are two sessions that are taking place this week that talk about the policy. There's a lot of talk about WHOIS policy and what we as a community and what we as Internet users should be doing about it and what types of access people should have and what types of access people need, balancing that with different privacy laws around the world.

If you have any interest in that, I strongly urge you to attend one of these concurrent meetings because that's how we roll here. So pick one, but don't go to both.

EPP.

UNIDENTIFIED MALE: Regarding the WHOIS protocol, it's mandatory that you have a Port 43 as a top-level domain registrar to provide this information?

ED LEWIS: Actually –

STEVE CONTE: No, go ahead.

ED LEWIS: Let's go back to what these tutorials are about because it's actually probably good to get this because we're starting to slip right now. The intention here in this tutorial is to talk about what are the tools in the tool box. Whether or not you use them is policy. You're asking a good question, but I want to bring it back to where we're coming from.

What we're describing in these slides are the tools that are out there, and registries over the years use these different protocols. In fact, with WHOIS, there's actually other forms in use today, one called RWHOIS, which we didn't mention, which was done in the RIRs, which had to do with we have a certain amount of data in our database you want to access through the WHOIS Port 43 protocol. But some of them you have to go to the ISP that owns the address space or is assigned the address space.

So in terms of technology, we have these different ways of doing things. The Thin and Thick question came up already, which is actually a good place where Thin and Thick are two approaches. One is Thick

says, “The registry is going to take everything back to the registry. I’m going to hold all the information about everything.”

In Thin, the approach is the registry only knows who gave it information, and they’re going to keep the piece. Now whether or not it’s a registry/registrar in all this, that’s getting into the policy space. So it’s just why your question is really pertinent.

You also mentioned Port 43, which I think is interesting because there are two kinds of WHOIS out there. There’s Port 43 and Port 80. Port 43 is what was defined way back when in the very early days. In fact, 43 is before 53, which is what DNS is. It was an earlier port assignment. Port 80 is basically just an HTML, a web interface, on top of that, for the most part.

So you said a couple interesting things in your question there that are worth bringing out.

STEVE CONTE: Before we move on to the EPP, are there any other questions about DNS?

ED LEWIS: Actually, I have one question, Steve. Can you help iron out this terminology? I’ve seen WHOIS systems refer to the entire registry, and the WHOIS protocol now, as something separate.

STEVE CONTE: WHOIS systems? Say that again.

ED LEWIS: In some of the things I've read, WHOIS refers to the entire registry. The term WHOIS has become jargon, and I see that in a lot of documents that float around this week. They talk about the WHOIS and the WHOIS is actually not the protocol, from what I can tell.

STEVE CONTE: The concept.

ED LEWIS: Well, sometimes it's used for the entire database. I don't know. Do you want to…?

STEVE CONTE: Oh, David Conrad wants to speak on that. David, okay, twist my arm. You can answer this one.

ED LEWIS: Here, you want two mics?

DAVID CONRAD: Two mics. Yeah, there we go. Do it stereo. This is actually a topic that SSAC took up a couple years ago and made the observation that WHOIS actually refers to a protocol, a service, and a database.

The ambiguity about when people say WHOIS was one reason SSAC put out a recommendation that people make a much more clear distinction about which of those three you're actually referring to. Instead of WHOIS as the database, it's the registration data. Instead of WHOIS the protocol, it's the registration data access protocol. I forget exactly what the acronyms are and the terminology that's used in the SSAC document because I'm about to pass out.

But that is a constant source of confusion. When people say WHOIS, what do you actually mean? In my experience, most times when people say WHOIS they actually mean the registration database.

UNIDENTIFIED MALE: Thanks, David. Nigel, did you still have a question? Let me grab you the mic. Hang on.

NIGEL CASSIMIRE: Yes, thanks. Nigel Cassimire from Caribbean Telecoms Union. Can you talk a bit about the routing of a WHOIS query under the Thick and Thin scenarios? Because I imagine it would start with an end user making a query. Where does the query need to go in the cases where you have [inaudible] Thick and alternatively a Thin registry situation?

STEVE CONTE: Thanks. It's actually not all that complicated. If the registry is Thick and the WHOIS database is Thick – and again, that means the registry houses all that data – when you do a WHOIS, kind of like when you're

doing a DNS look-up, there's a default set of answers. You have to know somewhere to get that questioned answered.

It knows that it needs to go talk to the registry first, regardless of if it's Thick or Thin. So my question is going to go to the registry first, and if it's Thick, the answer is going to come from the registry.

If it's Thin, the data inside that is actually referral data that's going to say, "Oh, you're talking about Conte.net? Go talk to" whatever my registrar – I don't remember offhand – but "go talk to them." So it's a referral, and then that answer comes back and you go and ask the referral point.

It's much like a DNS look-up. In fact, it's telling you, "I don't know, but go ask them. I know the person you should ask." It's not nearly as complicated as the routing that Alain threw up last session.

ED LEWIS: I want to add something to that, too.

STEVE CONTE: Okay.

ED LEWIS: In the protocols we talked about, the WHOIS and DNS, these are protocols. We think a lot about the server side but not much about the client side. I think it's actually important to break this down a little bit.

If you want to look something up, you're running a client. You're going to ask a question through the client software out there.

In both of these protocols – DNS, and I'll stick to WHOIS from here on out to be a little clearer – it's pretty dumb. It just says, "Open a connection, send a query, and get back the response."

Now, the software that you will be using – if you open up a terminal window and you're using a UNIX system or whatever is out there – that software is written by somebody somewhere. It's open source. It's been brought down.

People have spent time writing that open source software to be smarter than just, "Open the connection." In fact, when I type in "WHOIS something.some-top-level-domain," that client actually knows where to go by magic. Someone has already figured it out. They have a database somewhere saying, "Com is off at WHOIS.whatever.com," or somewhere else.

That software also, because there's no other specification here, has to be smart enough to know that when I get a response back from .com's WHOIS and they're Thin, I need to look for a special tag to know where to go next.

This is part of the problem: the protocol. It's so weakly defined that all this magic is actually encoded in open source software. You have to learn it from somebody else.

In this case here, the specification is so weak and the software out there has developed around it. Now, it works. The Internet works this

way because a lot of people just write code to make it work and we stick to that and it becomes de facto standards.

But it's a good question about how does this all get put together. There is magic happening because some people actually put through the grunt work to encode that and then distribute that to the systems, who then integrate it and so on, especially in the WHOIS. That's why WHOIS is so much of an issue today.

Other questions? Okay. So far we talked about DNS, DNSSEC, and WHOIS. For the most part, those are the oldest parts of registration systems. They go back quite a ways – DNSSEC a little bit less so. But WHOIS and DNS go back. The DNS exists. We have to manage the names in it and we have to have a registry, somehow.

WHOIS is there because in the early days, when you had a problem: someone's system out there was sending stuff across the Internet and you want to call them said, "Stop it." In a very cooperative time, it worked simply that way.

As time went by and as the desire to make registries become more technical beasts out there, to be more scalable to grow further, it became obvious that you needed a way to edit this database that's being maintained.

An EPP, an Extensible Provisioning Protocol, was not the first, but it was a significantly developed protocol in [inaudible] body to address this task.

EPP is what we call a business-to-business protocol. It's a protocol that assumes that the client and the server of the protocol have a pre-arranged agreement to allow me to do something. I'm going to try to add some information to a database or delete something from a database with anything else, all other things possible, billing and security built around that.

Its purpose is editing the database. The database is the heart of a registry out there. I can add or delete names. I can take them in and out of the database. I can add, delete, or modify the contact information, which a lot of times is the information held in the registration system. It doesn't have to be, but it is.

Transferring information from one place to another, who's responsible? Who's the client allowed to make the edits in there? Plus there are some other maintenance pieces to this.

The history? This was developed between 2000 and 2003. It came out of work done in COM/NET. COM/NET had its old protocol. I think it was called RRP (Registry-Registrar Protocol). That was not the direct basis for EPP, but it was the background work that went into the EPP documents.

They were developed within the IETF. The IETF had a working group that finished its work in 2003. Over the next part of the decade, the EPP protocol was refined a little bit more to be more standards compliant. Now it's actually a full standard protocol out there.

It has been mandated – this is a policy thing – for use in gTLDs and sTLDs by ICANN for its contracts. It has also managed to gain acceptance in the ccTLD world. There's no requirement that anyone use EPP. It's just proven that it's actually a useful protocol and it's basically rubbed off onto other operations out there. It's more extensible.

It actually kind of got out of control for a while and had a reconnection of all the parties. The IETF has convened a working group to help keep track of all of the extensions that are coming out for this protocol.

Keep in mind that EPP doesn't have to be an exclusive way to edit the database. It could be one way. It might be the exclusive one under policy. That's true for many of the TLDs out there, but not all of them. I worked in one registry where we had EPP running but we also had another way of editing the database because that was allowed in that policy regime.

Now, the EPP protocol is built on a TLS. TLS is a TCP with end-to-end security built in. So the client and the server talk to each other directly, not like DNS, where it goes through middle boxes. The exchange is XML, which is Extensible Markup Language, which is so ten years ago, apparently to some people It's an older way now to write data that was popular back in 2000 that nowadays kids don't like. I don't understand it, but they don't like it today.

The EPP server is going to be inside the registry. It's what's accepting all the edits from the registrars or whoever's allowed to edit the database, and that's how the database gets updated.

The EPP server on the registry side has a responsibility not just to do whatever's asked for but also may have to check policy decisions, like what names are allowed and does a name have to have a certain requirement for something? There are lots of issues that can come up there. Again, that's all in the policy space.

I'm going to go on to the next part. This past couple of protocols have all become at least ten years in place. They're pretty mature. RDAP is a new protocol. It was only minted last year by the IETF. It's called the Registration Data Access Protocol. David had mentioned it in his spiel over there.

This basically is an improvement on WHOIS. I'll go through describing what that is. What it is, it's the word up on top. It's a query/response means to inspect what's in the registration database. I don't care where the data is, whether it's in a Thick or Thin registry, if it's in an ISP for an RIR, the central registry, or somewhere else.

This is very much reflective of the idea of registration becoming a mature operational model. It's not at all restricted to domain names. In fact, it's not even really from the domain name industry. It came from the other side, the other side being the numbers registry.

The approach is to take HTTPS – the secure web protocol that's out there, the transport out there – and build on top of that. That gets rid of a lot of the underlying issues that we've had with the other protocols because they've been solved by somebody else.

The components. The server side has to have software to deal with the queries. It parses the queries. It has to have software to access the database, and software to prepare the response. Just a pretty simple three components there. It could be made complicated, and some people have done that, but it can be as simple as just, "Can I read what comes in over the network? Can I look it up in the database – SQL or whatever it is in the backend – and can I just format the response appropriately?"

The client side. Now, when I said about WHOIS, the WHOIS clients were really dumb. They open a connection and if you build something smarter into them, it's nice. The client is expected to be much smarter. There's more to it. It's a web browser API with specific abilities, including the ability to identify who the client is. "I claim to be so-and-so. I have a certificate to say I'm so-and- so." It can also do some authentication steps back and forth.

In other words, the idea of differentiated access came up. In the WHOIS protocol it wasn't possible. Now, differentiated access is totally policy. I have information about the home address for this IP address. Can anyone find it out, or just the police? That's policy.

In WHOIS, you can't do that because you don't identify who's asking. In this protocol, we allow the protocol to identify, "I am law enforcement and here's my credentials." On the other end, you decide whether that means anything to you.

Again, the building blocks are there. It's not whether you do this or not is not part of this. And there are many other parts of that. In fact, a

lot of the continuing work has been going into the policy over using this protocol.

Now the history of this. I read the section of the IETF charter in the last week, realizing that the initial work to get to RDAP, what it is today, which is basically a published protocol which doesn't have a lot of track record in a lot of place, but it's gaining it, started out with two of the RIRs being very dissatisfied with WHOIS. They tried WHOIS. They tried RWHOIS, which is Recursive WHOIS in their terminology, and they experimented with a web-based approach to get access to this data. A restful interface is one of the buzzwords they used.

They were very successful. They were very happy with the results. From that, they decided they were going to build RDAP coming out of this. It was to be a replacement of the WHOIS protocol. That was what it was trying to solve. That was the original problem.

It strongly emphasizes there's a commonality to number registries and domain name registries. At times, the IETF considered having separate solutions and talked like, "No, registration is a registration, no matter what it is. Whatever you're doing, it's the same function at this level."

On top of the HTTPS protocol, basically it said, "A lot of the problems we were considering have been solved," including web redirection that could be used here, authentication, and security. I could encrypt things. And a few other features. I think I have it on other slides.

The basic description: the queries look like URLs. They're certain-format URLs that say, "Here's what I'm looking up." The response

comes back in something called JSON, a Java Script Object Notation, which the kids seem to like. Between you and me, I can't tell the difference between XML and JSON. They look the same to me. But for some reason, JSON's cooler.

It's a formalized way of the responses coming back. That means now it's not like the old WHOIS class having to scrape information out of the responses and knowing that it's coming from comma and I've got to look for registrar colon or whatever or from ARIN I've got to look for this. There's actually a specified set of tags in the response that say, "If you want to go find more information, go here." It's much more crisp and clear that's way.

Actually, sorry. Crisp is actually a bad joke because after WHOIS, the IETF came up with something called IRIS, which is made by the CRISP working group. I didn't mean that reference. But it's out there now. That was a total slip of the tongue.

By the way, the RDAP was created by the WEIRDS Working Group. So that's the IETF for you.

Features. Why is RDAP so interesting to so many people? It has a defined data model. It has an expansion-friendly query and response format, meaning that it tells you how to say something without saying all you can say. It's not restrictive, but it has enough format that anyone could write a client that will be able to do some of the basic functions in there and interoperate with each other.

It allows for use of the non-ASCII character set. We hadn't talked about that in WHOIS, but the WHOIS protocol. Or that you could slip in non-ASCII characters into it. It really isn't built for that because those who wrote the old WHOIS clients only assumed ASCII and it wouldn't work too well.

In fact, the WHOIS protocol today, if you did put Unicode in there, it doesn't say how you encode it. Unicode's a nice way of talking about different character glyphs in different languages. But how you encode that, what the bit format is, it's not specific enough for that, whether it's UGF8 – there are a lot of other ones out there. WHOIS doesn't have that; in RDAP, that's handled.

Distributed data sources. Whether it's Thin or it's Thick, whether it's the ISP and the RIR, whether it's going around looking for different things, the protocol handles that just like web redirection handles looking at a website. When you go to some of the [inaudible] used for other content, it actually directs you to things. You may not know you've gone there. It's already taken care of.

Differentiated access is possible, but it doesn't say you have to do it. It's possible, and that can be thrown out to the policy folks. Say, "You define it. What's the differentiated access out there?"

Finally, when this protocol started, I went and talked to some of the guys behind us. I said, "Why will this succeed when the CRISP and IRIS thing (which I slipped and said) didn't succeed? What's the difference between these two? They're trying to do the same thing."

The answer is that this solution is more compatible with the current era of software engineering, so people will pick it up easier. Web technologies are a lot easier to adopt than what had come out with the earlier IETF version of a WHOIS replacement. I'll just leave it there.

This is the schematic. The RDAP client's on the left. It speaks RDAP to the RDAP message handler that the registry database is interacting with at a registry, whether it has the answer or has you go somewhere else.

I mentioned that the RDAP protocol gives you the ability to do things but doesn't say you have to do it a certain way. The session on Wednesday is going to talk about how ICANN is approaching this from a policy perspective called operational profile. Just because we know we want to use the RDAP and where it's going to be, it doesn't say how the data is going to be encoded. It's got to be specified somehow.

To continue on here, when you look at the registries out there, the registration field is getting more complex. Registries take on a bigger role. In an earlier slide I listed we had registries that were names and numbers. We also have land ownership and so on. Registration is a function which goes beyond just Internet technologies, so it takes on a heavier meaning now.

This database that we're talking about is very important. It generates our DNS. It exists because we want the DNS to be put together right. The WHOIS and RDAP let me look at it. EPP lets me edit this.

But what happens to that database? How can you make sure that this database doesn't get corrupted, get lost, get destroyed? We have operational ways to handle that. We have different ways of saving the database.

But what if the company that's responsible for the database completely vanishes also? Who's going to come in and rescue the disk with all these mappings that people have paid money for for these names? That's where data escrow came in.

The purpose of data escrow is that it's a third-party organization getting a copy of the registration databases and putting it somewhere that no one else can get to outside of a very strict set of rules. The business failure is one of the motivating reasons for having this come up.

Hopefully, if that registry has to be continued, can we keep it running even though we have these other problems? Can we get all that registration information from here to there through a third party when one of the parties may not be around?

An important thing here is that there are strict rules for access. ICANN uses data escrow for its contracted parties. ICANN doesn't have the escrow. It's somewhere else. ICANN can only go and get it under a certain set of rules.

The history of this. The IETF, one of the first things they do is hold Birds of Feathers sessions. These are: "We're going to look at this

problem. Is it worth having a working group?" I was a Co-Chair of that, so I saw this front and enter.

The IETF was bored with the idea because the data escrow is not interesting. It's not an exciting topic. And frankly, it's not really that exciting. It's just a matter of how do you copy what's in that database, whether it's MySQL or it's an Oracle database or whatever brands are out there? How do you put that somewhere? That's what it comes down to.

The IETF has not taken this on as a serious amount of work, but that doesn't mean that technically there's a problem with this. It just means it's just kind of uninteresting in terms of what the IETF would like to standardize. So there has been [continued] work on this.

Right now, there are two places that are defined in the ICANN context. One is there was a framework in an Internet draft. Internet drafts are kind of like formal documents that are out for public consumption, public reviews. It really hasn't progressed anywhere because the IETF really hasn't taken it on as a working group.

But also if you have to go into some of the other documents that ICANN puts out, there's the timing of actions. I can tell you how to copy your database to me – how do you take everything in your database and dump it out to me in bits? That's the first part of this.

The second part is, when do you do this? It makes sense to keep doing it over and over again because you make changes every day. So the timing of these actions are in another place in the ICANN world.

It comes down here in an XML version of the database. It's compressed/encrypted and by ICANN's current set of policy rules or whatever you want to call them, a full dump of the database is done every week on Sundays, and every day the incrementals are adding to the escrow.

This is a little picture of what happens where the registry operator, who's on your left – it never works when you're speaking. It sends the keys and notification to the agent – escrow agents are some companies out there; there a few of them that are accredited – and sends to them all of the deposits.

To ICANN it gives notifications and the keys for access when it's needed, and that's what the registry sees as far as this protocol.

Any questions about that? Data escrow is kind of – yeah, Ashely?

STEVE CONTE: I think we have people online.

[ASHLEY HEINEMAN]: Based on what you just said, it sounds like the data escrow gets populated once a week. So if a TLD goes under, there is still the potential that within that week period any new registrants could be lost? Is that a fair assumption?

ED LEWIS: I believe with the ICANN it's more on a daily basis because it has incrementals. It could happen that they may lose 24-47 hours' worth of updates if something goes south like that.

I'd just see how that would be handled. People would probably complain about it if they're valid registrations. I'm not sure if it's actually specified to that level of detail and I don't believe that it's ever been tested. So I don't know a good answer.

Any other questions? Is data escrow really exciting? Go to the IETF and tell them we want to work on this, okay? Please. I'm just kidding. But you can go to the IETF and talk to them.

Oh, I should say this too. Data escrow: there really isn't a protocol name for it because the IETF doesn't have a working group with funny names yet. So it's just called data escrow.

Yes?

UNIDENTIFIED MALE: More of statement. There's two types of data escrow. There's the registrar data escrow and registry data escrow, and they are two very different kettles of fish.

ED LEWIS: Okay. You said that's a statement. Yeah. Actually, that's interesting that you say that because that's why the IETF would have been a good place to discuss this because they want to be able to make, "What can I do to make it common?"

The worst thing in the world is to have two standards for something. You'd like to have one so they can combine it. They can exist as they are, but in a sense that the more energy there is to solve this problem – and this actually the kind of thing that could come up – it's not necessarily just about policy here, but technically if you want to have one solution, the IETF is probably the best place to debate that. You just have to make sure that there's energy/budget to go to the IETF and say, "We want to do the work in the IETF."

I would say that if it is something that is causing operational pain, it's probably worth approaching. I'm giving you a very fishy answer. Go to the IETF and tell them to start working on it.

And the IETF has its own dynamics. I don't want to get into that. It would take too long to describe all that.

Any other questions? Okay. I just thought data escrow would be a tough sell anyway, so I didn't have high hopes for them.

The next one: the trademark clearinghouse. I'm going to tread into this one lightly, partly because it's got a little more work behind it technically, and I didn't have time to really put it all into these slides. So I'm going to concentrate basically on what the trademark clearinghouse means to the registry. I'm going to try just to stay in that area because the trademark clearinghouse is an interesting topic to me.

I'm actually not sure who uses it, and ICANN has put it in place. Its chief role, as far as I can tell, is so that people who own trademarks

can tell ICANN, "I have a trademark, and if someone registers a domain name that might be in conflict with that trademark, something is got to happen." "Something" is a loose word.

It's an interesting way of looking out for certain interests out there in the world, combining the outside world with the Internet. The domain name is the Internet world. Trademarks are the outside the world. Here's a nice little example where they're starting to collide with each other.

I'm going to limit what I'm saying here to just what the registry has to do in this area, mostly just to advertise it actually exists. It's an impact on our registries.

To get into this, though, there are two phases that are important that actually are mentioned in the documents that talk about TMCH: the Sunrise and trademark claims.

This again is more policy but helps to understand the phase of the protocol. Sunrise refers to a period of time – people can correct me if I'm wrong on this because I'm probably the least educated person on this in the room – before a TLD goes out and says, "Anyone can have any name they want," they offer names to the trademark holders, and the trademark holders are registered through another process which I'm not explaining. There's a period of time in the opening of a TLD where if you have a trademark name, you can get it before anyone else has the chance to go get it.

During trademark claims is another period of time where, while it's in place – I don't know how long it takes; I don't think it's specified exactly how long it takes place – every time you register a domain name that happens to match someone's trademark registration, notices are sent out. It's not blocked. It's not prevented. It's not action automatically happening. But they're saying, "Hey. Someone just registered this name, and you just registered a name that's a trademark." It goes out there. These two are different phases

In Sunrise, the registry gives a list to the TMCH, the Trademark Clearinghouse, as an agent – I think there's more than one; I'm not sure if there's more than one or not – a list of domain names that are registered so that the trademark clearinghouse can go and see if there are any conflicts or any matches between the two.

The registry receives from the TMCH – the wording here looks a little funny – the names that are no longer listed as trademarks. It doesn't give it the list of trademarks. It gives the names that have been revoked from that because earlier in the process there was a list given of all the trademarks to watch out for and they rescind them over time.

So in the Sunrise period, names get unregistered from the trademark list. Meanwhile the registry tells the TMCH everything that's going on.

During the claims, things have changed now. The registry supplies the trademark clearinghouse the list of domain names registered that match the list that they're looking out for. Not all the names that are

getting registered, but only if there's a match between what's registered by somebody against what they're told to look out for.

The registry also receives from the trademark clearinghouse the labels that are registered corresponding to a pre-registered trademark. Actually, it's the latter use of the word "registered" that triggers the fact that we're talking about domain name registries and trademark registries are different registries.

This is a diagram of this showing during the Sunrise period, the revocation list goes across from the trademark clearinghouse, claims that the domain list is going over, and then on the way back there are names that match something, depending on which claims period I'm in.

Questions about trademark clearinghouse? Or does anyone want to correct what I said that was wrong?

UNIDENTIFIED MALE: Who operates the TMCH? Is it ICANN?

ED LEWIS: No. It's a third party. I know for a fact. I don't know if there's one or more of them. Somewhere downstairs I heard someone saying we are an official or the official trademark clearinghouse operator.

UNIDENTIFIED MALE: Are they using RDAP to communicate with the registry?

ED LEWIS: Well, no. I'll say I don't think anyone's using RDAP right now.

UNIDENTIFIED MALE: Okay.

ED LEWIS: But I believe if you want to know what they're really using, there are a couple Internet drafts that describe the whole system pretty well with some good ASCII art – actually, explainable art. Also, there is something in the specification documents that would help answer your question.

UNIDENTIFIED MALE: A correction. I believe the RIRs do have production RDAP servers.

ED LEWIS: Yeah. I'm sorry. I was thinking of in the ICANN. Yeah. Yeah, the RIRs do have RDAP. That's all separate from this.

UNIDENTIFIED MALE: It just clicked that the TMCH is another registry, right?

ED LEWIS: Yeah. It basically is another registry, yeah. It's not a domain name registry, but yeah, it's a registry and it has its own structure.

Another question?

STEVE CONTE: Any other questions? Anybody?

ED LEWIS: Over here. Oh. Okay, okay.

UNIDENTIFIED MALE: Thank you. My question is, I'm seeing the process of the trademark. Okay. Is it basically automated? Then, if it is automated, which registries basically use them that you might know? Because I'm looking at it as a ccTLD perspective, how it could be implemented in the ccTLD environment.

ED LEWIS: I myself don't know of any ccTLD that does this. They may, but I don't know. ICANN has it in its agreements with the new gTLDs – what I call the Class of 2012 gTLDs. I don't know if they extend back to the older TLDs, what I call the Class of 2004, 2003, or 2000s. So I'm not sure. Basically, a registry may have to use it based on its agreement with ICANN, which may have changed over time.

As far as I know in terms of the automation part of this, from what I understand, this does not prevent registration of names, but it sends out notifications of events that may become a trademark issue.

So that's [inaudible] I'm going to be about my answer.

STEVE CONTE: A question over here.

UNIDENTIFIED MALE: More of a sales pitch. We've got our own system called [Direct] Verification System, which is based on the TMCH systems. We use it for moderation as well as for mark verification. We used it in our ccTLDs, which we've launched recently as well.

ED LEWIS: Okay.

UNIDENTIFIED MALE: I'm from .za with [inaudible], Durban and Cape Town as well.

ED LEWIS: Yeah. Okay. Yeah, I'm just not familiar with this. Yeah. Okay.

STEVE CONTE: So not just trademark. Any questions about anything that we've spoken about today? Alright.

ED LEWIS: So now you can all build a registry from scratch.

STEVE CONTE: Maybe registry. Any other questions or comments or sales pitch or something?

UNIDENTIFIED FEMALE: This is not the most intelligent question, but if the WHOIS protocol is now moving to RDAP, will we just no longer refer to it as a WHOIS protocol? Or is that going to remain the nomenclature?

ED LEWIS: It's kind of like when I said I tape my TV shows today.

STEVE CONTE: Yeah. Or use a Kleenex.

ED LEWIS: I have a feeling that it'll be a long time before WHOIS gets wiped out. Because of the SSAC [inaudible] that David mentioned, too, is that WHOIS has become jargon for a number of things. In fact, I could see the WHOIS protocol and RDAP protocol co-existing for some time. There's no reason why the two of them can't both be ways to look at the database.

I've seen some people do napkin writing of how to make the two work together, about policies for differentiated access and all that. So I've seen people considering that.

STEVE CONTE: Denise?

[DENISE MICHEL]: Just a quick follow-up on that. The Board has initiated a policy development process, has asked the GNSO to start a policy development process, to develop a next generation WHOIS, essentially. Scott Hollenbeck and several other people served on a working group that developed a model and proposed policy structure for the GNSO to consider. So I suspect that the phrase "WHOIS" and "WHOIS replacement" will be around in the ICANN world for at least the next, oh, five years I guess?

ED LEWIS: Are they going to have an event this week? On the calendar.

DENISE MICHEL: I'm sorry. Who?

ED LEWIS: The effort you're talking about with the PDP.

[DENISE MICHEL]: No, the next step in that is that staff will post probably in about a week or so after Buenos Aires a preliminary issue report for public comment. Then that will kick off the PDP process.

ED LEWIS: Okay.

STEVE CONTE: As long as we have the fora, we will be calling it WHOIS.

ED LEWIS: Yeah. One of the things I want to mention, too – I should have mentioned earlier in DNSSEC – in sense of trying to highlight events during the week, there's a DNSSEC workshop on Wednesday that applies to the DNSSEC protocol here. That's probably why I was asking if there was an event advertised.

UNIDENFITIED FEMALE: Are these slides going to be online?

STEVE CONTE: They're already online. If you go to the main agenda and click into the session, the slides are already posted.

DAVID CONRAD: Excuse me. I also believe the transcript of this will be made available, and also translated. I believe the translation team will be – yeah, the audio recording will be transcribed and then translated into different languages. Which is sort of the definition of translation, but never mind.

Any other questions? Good. In case anybody doesn't know me, I'm David Conrad. I'm ICANN's CTO. This set of tutorials was actually an initiative that was suggested that ICANN should do. Within my group, we decided to take this on. I hope it was useful and productive to you.

We will be doing similar tutorials likely in future ICANN meetings. We're planning on doing something similar to this in Dublin. If you found this valuable, please tell your friends and neighbors.

If you have any suggestions or comments, please let staff know. Let me know. I'd be interested in hearing what people had to say, whether they found it valuable, what could be improved, what was spectacular – all of that, anything that you might want to tell me. I'm David.Conrad@ICANN.org.

With that, if there are no other questions, I'd like to thank Ed and Steve for providing this tutorial and thank you for attending.

Oh, and there's a very little nice deck out there. Nice view. It's a pretty day and everything. Should have gotten cocktails or something out there. Maybe next time we're in Buenos Aires.

UNIDENTIFIED MALE: More people that way.

DAVID CONRAD: Exactly. More people.

[END OF TRANSCRIPTION]