ICANN84 | AGM – SSAC Presents: DNS Runs on Free and Open-Source Software
Tuesday, October 28, 2025 – 13:15 to 14:30 IST

KATHY SCHNITT

Thank you. Hello, and welcome to the SSAC Presents: DNS Runs on Free And Open-Source Software. My name is Kathy, and I am the participation manager for this session. Please note this session is being recorded and is governed by the ICANN Community Participant Code of Conduct, the ICANN Expected Standards of Behavior, and the ICANN Community Anti-Harassment Policy.

Please observe the following guidelines to participate in this session. I will also post them in the chat for your reference. Only questions in the Q&A pod will be read aloud during the session as time permits and when directed by the chair of the session. If you wish to speak, please raise your hand in Zoom or otherwise as directed. When speaking, please state your name for the record and speak clearly and at a reasonable pace. And with that, I'm happy to hand the floor back over to Tara Whalen.

TARA WHALEN

Thank you, Kathy. As Kathy stated, My name is Tara Whalen. I'm the vice chair of SSAC, but today I'm coming to you as a member of the work party for SAC132, The Domain Name System Runs On Free And Open-Source Software or (FOSSS), which we're pleased to present to you today. And I will be giving the presentation, along

with one of the work party co-chairs, Maarten Aertsen. I know we have many of the work party members here in the room, and so we hope this will be a very active discussion among all of us today. Next slide, please.

Here's what the agenda looks like for today. We will be talking about how the DNS works, the FOSS prevalence in the DNS ecosystem, then some information about understanding FOSS, and closing out with some guidelines for policymakers and the ICANN community. The goal of this report was in order to enable development of effective, nondestructive policy that strengthens rather than undermines the FOSS ecosystem that is critical to the stable operation of the Internet. And our approach is to build understanding in layers, so you can see why the way that we think about software for DNS infrastructure needs to be different.

Now, during this session, this is designed to be interactive, and we do encourage questions throughout the presentation, rather than just having a lecture with Q&A at the end. So please do jump in with questions, and Danielle will ably handle the queue for me. Thank you. Next slide, please.

We're going to start today with some introductory slides to build a basic understanding of the technical components of the DNS. Next slide.

All right. We have a look around to this diagram. On the light blue section, we have the domain name registration component, and this is the administrative and contractual framework for acquiring

domain names. And this is where most of the ICANN community operates—the registries, the registrars, and the registrants. The relationships in this layer are well understood, and they're defined by contracts. So the ICANN to the registry, registry to registrar, and registrar to registrant, and these established clear lines of responsibility between actors.

If you look at the green section of the DNS infrastructure, this layer represents the technical plumbing that makes domain names function. It's run by many independent organizations with different operational models. The DNS infrastructure layer is governed by operational relationships and technical standards. The information lookup or the queries is mostly not monetized, though a user does not have a contract to perform a lookup. And as the diagram shows, each registry on the left has a corresponding DNS infrastructure on the right.

Then if we move to contents and services, the orange and dark blue sections illustrated, this is the top layer that end users interact with. This would include websites, e-mails, various applications. That's not our focus today. A note, of course, registering a domain name is not the same thing as creating a website and publishing a domain in the DNS so that it can be found is not the same as publishing website content. Next slide, please.

A disclaimer for this. This is a simplified flow to illustrate. So, consider the scenario of what happens when you type www.icann.org into your web browser. Step one, you have the

query with the application, so your device, your laptop, your mobile phone, etc., needs to translate the human friendly domain name into a machine readable IP address. And to do this, it sends a DNS query, which then moves us to step two, which is the helper, the resolver, where the query goes to a recursive resolver which acts as a helper to look up the answer. Now, this is typically provided by your Internet service provider, your ISP, your company's IT department, or a public service such as Google's 8888.

So then at step three, the lookup or the hierarchy query, the resolver queries the DNS hierarchy to find the corresponding IP address. Now, there may be a fast path where it first checks if the answer is already stored or cached from a recent request or it may do the full lookup. So if that answer is not cached, it performs the complete step-by-step lookup process. At step four, we get the answer or the connection. The IP address is returned to your device, and your browser can now connect to the www.icann.org web server. The query process happens thousands of times a day on a typical device, and each lookup is usually completed in milliseconds. Next slide, please.

Note that the DNS namespace is structured as a hierarchy like a pyramid. So the root at the top of the pyramid, this has their 12 root server identifiers, the A route through M root, that are operated by 12 independent organizations globally. Its primary function is to know the location of the top-level domain or TLD name servers.

The next level down on the hierarchy is the top-level domains, or TLDs, that include two main types: the generic TLDs or gTLDs such as .com, .org, or .net, and the country code TLDs or ccTLDs such as .uk for the United Kingdom or .jp for Japan, and on with other countries. And then there's the individual domain names, and this is the bottom level containing specific names like icann.org. This hierarchical structure is what allows the system to scale to billions of names. And there are operators and software at each level of this pyramid: for registering domain names for the registration infrastructure and for publishing information in the DNS, the DNS infrastructure. Next slide, please.

Every component in the DNS ecosystem runs on software, which must be continuously written, maintained, updated, and secured. We'll go through a few examples. Next slide, please.

You'll note in the circle, root servers and TLD operators use DNS server software. Next slide.

We'll note the resolvers are using DNS resolver software. Next slide.

The registry operators rely on database and application software. And you can see the big circle, software powering everything. So this led us to ask critical questions about the software supply chain. Where does the software come from, and who builds and maintains it? We feel the answers are important for a field that is increasingly regulated and policymakers have a need to know, even if it may not be surprising to those of us in the technical community.

A moment for audience participation. How many of you know what DNS or registry software your organization or your TLD operator uses? Oh, look at all those hands. Similarly, how many of you would know who maintains that software? All right, we have a couple of people. Okay. A small number of folks. Glad some of you have the answer to that. That's great. Next slide, please.

We've established that software is critical. But those questions on the last slide of where does the software come from, and who builds and maintains it, that's what leads us directly to our next section. So let's take a look at the answer that the SSAC found, which brings us to the core research of our report, the prevalence of FOSS in the DNS ecosystem. So this is the core finding of SAC132. You'll note here that DNS is built and sustained on free and open-source software. This isn't advocacy, though this is not us saying that FOSS is better. This is a statement of fact about how DNS infrastructure is actually built. Next slide, please.

We surveyed all 12 root server operators to identify the software they use, and there is a definite dominance of FOSS. So at least nine of the 12 root server operators use free and open-source software exclusively. Common FOSS implementations include BIND 9 maintained by Internet Systems Consortium, NSD maintained by NLnet Labs, and Knot maintained by CZ.NIC.

And there's some hybrid use here as well. So even operators that are using some proprietary software often run FOSS alongside it for diversity and redundancy. The key takeaway here is that this is not

a small percentage. FOSS is the overwhelming majority in the root server system. So we'll note, for example, that BIND 9 is the oldest and most widely deployed DNS software with its origins in the 1980s. NSD is known for its high performance and efficiency as an authoritative only server. Knot is a more recent but increasingly popular implementation from the operator of the .cz TLD, and many of these projects are incredibly stable and have been maintained for over 25 years. Next slide, please.

We analyzed the largest operators that provide authoritative DNS services for top-level domain registries. We had several key findings: 9 of the top 10 TLD service providers use FOSS for their DNS infrastructure. And for country code TLDs, or the ccTLD specifically, 20 out of the top 25 operators providing authoritative DNS service for countries and territories use FOSS DNS software. These 20 operators collectively provide DNS service using FOSS for 234 unique ccTLDs. And 23 out of 25 ccTLD operators that do not use internationalized scripts rely on FOSS.

So the scale is massive. These are not small experimental TLDs. They are major service providers responsible for millions of domains. For example, Nominet who operate the .uk TLD, which is one of the largest ccTLDs, use FOSS name server components. CNNIC that operates .cn in China is one of the largest TLDs in the world. And the Public Interest Registry, PIR, operates the .org TLD, and all of which are making substantial use of FOSS. Next slide, please.

When we look at registry systems, we see two main models in use. So there's Model 1 of entirely FOSS. There are complete end-to-end registry platforms that are entirely free and open-source software. Now, this is an important point even if this is not the model used by the very largest registries. Some good examples include the FRED platform, which is used by at least 12 country code TLD registries, and Google's Nomulus platform, which they use for their own gTLD registries. This shows FOSS is a viable complete solution for critical infrastructure, not just a minor component.

Absolutely, I'll stop for a question.

DANIELLE RUTHERFORD All right, we've got a question from Satish. Satish, you want to go ahead and unmute and ask your question?

SATISH BABU Thanks very much. I had two quick questions. In the previous slide, you mentioned a rider for those domain names which are not internationalized. Why is that?

DANIELLE RUTHERFORD I'm sorry. Could you please speak a little bit closer to the mic? We're having a hard time hearing you.

SATISH BABU I was saying that in the previous slide, while mentioning these numbers, you said those that are not handling internationalized

domain names. So that seems to be a kind of a qualification. So is there any problem inherent in FOSS software not able to handle internationalized domain name? That is one.

Second point is, these numbers are a snapshot in time. Do we have any trend of how many are migrating into it or out of it so that we are aware, we could be kind of better informed? Thank you.

MAARTEN AERTSEN Hey, Satish. This is Maarten Aertsen speaking.

DANIELLE RUTHERFORD Closer to the mic.

MAARTEN AERTSEN Oh, I need to eat it. That's great. This is Maarten Aertsen speaking. With respect to your question about IDNs, no, there is no such limitation. We collected data, ran the statistics, and then we just noticed this anomaly, in the sense that there are a couple of IDN ccTLDs that happen to be run by one of the major providers that is using proprietary software. So it's more of a statistical anomaly, and we list the conservative number here, which is both IDN and ASCII labels, but it was interesting to us to see that it's even larger than when you count for that difference, but it has nothing to do with the software.

Now, with respect to the second question, we only have point in time data. So we have some academics that were part of this

research. One of them invited guests, and he kept us honest. So, in the report, you'll find a sentence that says that we know this data for today. I think some of us maybe even in the room might have something to say about what they think about the past, but we didn't collect that data. So I guess one data point that is not in the report, but is worthwhile to consider, is that the maintainers of the popular name server implementations, they didn't survive to this day because no one was using their software. So, in that sense, it's quite likely that it was popular before. But yeah, we don't have that data. Thank you.

TARA WHALEN Thanks for the questions and giving us the chance to put some clarifications out. So I'll go back to the slide on FOSS and registry systems, number 16. That would be forward to 16.

DANIELLE RUTHERFORD Other direction, Kathy.

TARA WHALEN Yes. Perfect. Thank you. So we looked at the Model 1 of the entirely FOSS systems, but the second model is proprietary on FOSS. So this is more common for the largest registries. This is proprietary custom-built systems. But note that these proprietary systems are not built from scratch. They are proprietary integrations built on top of a foundation that is overwhelmingly FOSS.

So, for example, all major registry platforms that we surveyed rely on FOSS components. There are databases like PostgreSQL or MySQL, web servers like nginx and Apache. There are application servers that run on various Java frameworks that are often FOSS. And there's FOSS in many of the monitoring and testing tools as well. And of the 10 major registry platforms we surveyed, all but one incorporates significant FOSS components. So even when the top-level business logic is proprietary, the foundation is FOSS. And the proprietary value lies in the integration and business workflows, not in the low-level infrastructure components. A number of registry operators maintain registration infrastructure that is entirely FOSS like CZ.NIC's FRED and Nomulus. Next slide, please.

We have a case study. Cloudflare is a 1.1.1.1, and this is one of the world's largest public DNS resolvers. The core resolver code is proprietary and closed source. However, it is completely surrounded by and dependent on FOSS. It's written in Rust, which is a programming language with a FOSS compiler and library, it uses Tokio, which is a FOSS asynchronous runtime library, and it runs on Linux, which is a FOSS operating system. All the surrounding observability, logging and monitoring tools are predominantly FOSS.

To quote from SAC132, "Cloudflare's proprietary software behind 1.1.1.1 operates at scale thanks to FOSS, not in spite of it." And Cloudflare is very public about this and contributes back to the FOSS ecosystem they rely on.

So this is the reality of modern software. Even when the top layer is proprietary, the foundation is collaborative, open source development. But such a model creates a complex, interconnected supply chain, which has major implications for how we think about security and policy. Next slide, please.

We'll note that resolver infrastructure was the hardest to measure, so instead of deployment, we looked at what's available on the market. The resolver ecosystem is vast and diverse with millions of resolvers worldwide. So, local resolvers are serving about 80% of users. These are operated by ISPs, enterprises, and universities. The software is predominantly FOSS, BIND 9, Unbound, PowerDNS Recursor, or the Knot resolver. Now, there is a commercial market as well, but if you examine what they offer, it's almost always FOSS inside. And there are Cloud platforms as well and major platforms like Microsoft Azure, Google Cloud, and Amazon AWS operate massive DNS resolver infrastructure. At least four of the biggest hyperscalers rely on FOSS for their DNS resolution. Others use FOSS libraries as components within proprietary resolvers. And we also have the public resolvers. So Quad9 runs on Unbound, which is FOSS; DNS4EU, which is an EU initiative, runs on the Knot resolver, also FOSS; and as already described, Cloudflare's 1.1.1.1 has proprietary cores that are built on FOSS foundations. So you'll note a substantial presence across the entire resolver ecosystem. Next slide, please.

This is what we found across the entire DNS ecosystem. For the root servers, at least nine of 12 operators use FOSS exclusively. For the

TLD providers, 9 of the top 10 use FOSS. For the registry platforms, 10 out of 10 incorporate significant FOSS components. For the resolvers, most software across ISPs, Enterprises, and Cloud is FOSS. And for proprietary systems, even these are often built on FOSS foundations.

To quote from SAC132, "In the most critical parts of DNS infrastructure, FOSS is the norm and proprietary software is the exception." This isn't an exaggeration. This isn't advocacy. This is a rigorous finding from SSAC research, which is that the Internet's core addressing system runs primarily on software that is collaboratively developed, often by volunteers, and freely distributed. So at this point—next slide, please—I will hand the microphone over to my co-presenter, Maarten, to give us some understanding about FOSS.

MAARTEN AERTSEN Thank you, Tara.

DANIELLE RUTHERFORD Closer to the mic.

MAARTEN AERTSEN Oh, even closer. Let me try and put it in my mouth. Thank you, Tara. So we'll now turn to try and understand what FOSS is, because the numbers on that last slide are clear. FOSS is the dominant reality of the DNS. This is in effect with massive implications. But to understand why this is so important for policy, we need to be very

precise about what FOSS is, and just as importantly, what it isn't. It's not just free software. It's a completely different model for building and maintaining critical infrastructure. Let's dive into that in our third section.

FOSS stands for free and open-source software, and "free" here refers to freedom, freedom of speech, not price. Free Guinness after this talk. I said not there, but yeah, I do appreciate if you will have… Anyway, FOSS is defined by four essential freedoms granted by its license. Use the software for any purpose with no restrictions, study how the software works with no restrictions, share and redistribute copies of the software to anyone, and change the software to fit your needs and release copies of those improvements.

These freedoms are legally granted by software licenses. For example, the GPL, the BSD license, or Apache. If any freedom is restricted, it is proprietary software, not FOSS. And these freedoms create a fundamentally different ecosystem for software development compared to the proprietary model. While there are philosophical differences between free software and open source, this report uses FOSS to encompass both concepts centered on the four freedoms. Next slide, please.

The traditional proprietary model is where a vendor sells a license or subscription to a customer. The vendor can contract with sub-vendors for components, and there is a clear entity to hold responsible, and there is legal recourse. For example, suing for

breach of contract. From a policy view, you can regulate the vendor through this chain of contracts. Now, in the FOSS model, a maintainer, an individual team or foundation, publishes software, an operator downloads and uses it. And there is no contract by default, only a license that grants these four freedoms. The license typically include as is or no warranty type language, and no one is legally required to fix issues in this software. In some cases, there is even no single responsible legal entity to hold responsible because there is FOSS that is created by groups of individuals.

So, in FOSS there is often no contractual relationship between the creator and the user of software. A volunteer can write code that a TLD operator ends up using with no payment contract or ongoing obligation. This is why traditional supply chain regulation fails for FOSS. You can't impose contractual obligations where no chain of contracts exist or hold a vendor liable when there is no vendor. Next slide, please.

Roles in the FOSS ecosystem are fluid but fall into three main categories. There is the maintainer—guides the project's direction, reviews contributions, and ensures quality. This can be a volunteer or someone employed by a foundation or company. For example, ISC employs maintainers for Bind 9. There is contributors. These are people who submit bug fixes, new features, and documentation. Again, this can be anyone, a university researcher, an operator, or a developer working for a company. So, contributions are reviewed by maintainers before being accepted. Now, finally, there is the operator or user. This is the entity that is

deploying the software in a production environment. For example, a TLD operator. They can provide feedback, report bugs, and they are responsible for keeping their systems running and updated.

These roles are collaborative. There need not to be a contractual chain or required payment flow, and many people in the DNS community play multiple roles, operating infrastructure while also contributing bug reports and sometimes even patches. Another example, Bind 9 is maintained by ISC, but contributions can also come from operators worldwide. ISC then reviews and integrates these, and TLD operators then update their deployments. Next slide, please.

The FOSS model has a different risk profile than proprietary software. So let's talk about some inherent risks of FOSS in general. Now, most FOSS, if you look at statistics, rely on single, unpaid volunteers. Because they are volunteers, motivation plays a large role. So if you run out of motivation, you can burn out, and this leads to project abandonment. Now, as we said before, FOSS inherently does not have guaranteed support by default, so no one is legally required to fix bugs, so operators must take responsibility to retain development expertise in-house or purchase separate support contracts.

This brings us to a financial fragility aspect, because funding is decoupled from use, so you get into the territory of free riding, where an organization relies on software but does not contribute back either feedback, patches, or money. So where the maintainer

is employed by a small initiative or an organization, small company, and they pay maintainers to do this work professionally, they are vulnerable to funding shocks, in particular, funding shocks resulting from new regulatory burdens, for example, when an existing funding model is threatened. So these risks are real, and they are inherent in the FOSS model. They are trade-offs for its strength. So regulation that doesn't understand these trade-offs can make these risks worse, not better. Next slide, please.

Let's talk about some strengths. There is the transparency and collaborative security angle. So the academic and operator community maintain a strong, active culture of openly scrutinizing FOSS DNS software, and this has been going on for decades. You can find papers in the '90s pointing out flaws in BIND and subsequently getting them fixed. And flaws or improvements are openly discussed and tend to be fixed because of these discussions. We surveyed operators and they appreciate the ability to diagnose, verify, and expedite patching of vulnerabilities, precisely because the supply chain is transparent. So if the name server software they use integrates a certain component and it is known that this component contains a vulnerability, they can patch it themselves, because they know what's in there, as opposed to when you have a proprietary name server and you have no control over when your vendor provides you with a patch.

Now, for the DNS specifically, there is also the stability and long-term support angle. So, it is the case that a number of popular DNS projects are supported by long-lived and stable organizations. So

in the report, we treat ISC, NLnet Labs, CZ.NIC, and PowerDNS that have all been maintaining their software for 20+ years. So this type of lifespan provides a proven track record that operators trust. And it is a contrast with FOSS in general, where really important software can be in a situation where there's no organization backing it, it might just be a collection of volunteers. And in the DNS, it's exceptional for these organizations to exist.

Now, finally, there is the angle of operational resilience through diversity. Because there are multiple FOSS implementations, operators can run different software concurrently and reducing the single point of failure. It also lowers the barrier for organizations to run their own infrastructure, preventing concentration risks. Next slide, please.

Now, let's review security. What doesn't determine security is whether software is open or closed in its source. Also, what doesn't determine security is whether developers are paid or volunteer, whether it's from a commercial company or a nonprofit. What does determine security is rigorous code review and comprehensive testing, a well-exercised vulnerability disclosure process, and an active long-term maintenance with sustained resources.

So, in the report we have this quote saying, "The security of any software project is determined by the quality of its development and maintenance processes, not the visibility of its source code."

A question that is asked a lot is is FOSS more or less secure than proprietary software? And the answer is neither. Security is

determined by the quality of the development process and not the licensing model. So the policy challenge is that FOSS has a different risk profile. You cannot copy-paste regulation from the proprietary world and expect them to work. Next slide, please.

The traditional approach to regulation is where you require an operator to meet security standards, possibly imposing them by law on operators of critical infrastructure, and then hold them liable, and the operator, in turn, tries to hold their suppliers liable. So these rules would end up being enforced through procurement and market access. This fails with FOSS for a number of reasons.

The first problem is that FOSS is often provided for download at a zero cost, so there are not necessarily contracts underlying the use of FOSS. There is no supplier relationship in the traditional sense, and there might not be an operator vendor contract for software to attach obligations to, or there's even no vendor to regulate, just a volunteer. So traditional supply chain regulation is designed for a vendor-customer model that assumes financial transactions. Because software does not have an inherent material cost per additional copy, and can therefore be provided for near zero cost, neither financial transactions nor contracts are guaranteed to exist. Now, imposing a compliance burden on a volunteer maintainer could cause them to abandon projects due to the legal risk, switch to proprietary licenses to limit liability, destroying the benefits of FOSS, avoid regions or jurisdictions to avoid regulatory

risk, and the result is that the software that everyone depends on becomes less available and less secure. Next slide, please.

So, if we get this wrong, there is a number of things. Risk one, infrastructure instability. Critical projects are abandoned by maintainers, forcing operators to use less suitable alternatives and potentially causing service disruption.

Risk two, innovation suppression. Volunteers stop contributing due to legal risks, leading to less software diversity, greater concentration risk, and vendor lock-in.

Risk three, digital autonomy erosion. So, smaller organizations, for example, a developing country ccTLD can't afford proprietary software and becomes dependent on a few large commercial providers, losing technical sovereignty.

But if we get it right, we'll be in the right hand of this slide with a strong, sustainable FOSS ecosystem. Maintainers are supported, not burdened. Projects thrive and security improves. We get to secure and stable DNS infrastructure with proven software continuing to serve billions a resilient global Internet and reliable operations. We have a continued innovation, new projects emerging, competition thriving, multiple implementations flourishing. And finally, we can have an accessible infrastructure for all. All organizations can participate, digital sovereignty is preserved, and local innovation enabled. Next slide, please.

The last slides make the stakes crystal clear. We all want the positive outcomes on the right, the stable, innovative, and sustainable one. The question is how do we get there? How do we create policy that supports this ecosystem instead of breaking it? And that is the entire point of this report, and it's what we'll cover in our final section, guidelines for policymakers and the ICANN community. Next slide, please.

Major jurisdictions like the US, UK, and EU are developing FOSS aware policies. The first approach is to allocate responsibility appropriately. We've seen this in the US and the UK. This approach places obligations on commercial developers or commercial deployers, not on open-source maintainers. So a volunteer FOSS maintainer publishes software freely with no regulatory obligations. A commercial entity, say, an integrator that puts a piece of FOSS inside their commercial product would be subject to requirements, and they are responsible for final product. A infrastructure operator that is a user is responsible for its own operational security, patching, etc., etc.

Approach two is incentivize sustainable maintenance, and we see some of this in the EU Cyber Resilience Act that introduced the concept of an open-source software steward. So a steward is an organization, for example, a foundation, that provides sustained professional support to a FOSS project. So it's a bit early, but this may solve part of the free riding problem by creating a channel for commercial users to fund maintenance, and this model can fund

critical projects without burdening the volunteers that might be behind this steward.

Now, a third approach that we've identified is to adapt supply chain concepts. The European Union also has a NIS2 Implementing Act which is basically regulating critical infrastructure in Europe, not just Internet, but also like water and stuff. And here the NIS2 itself required managing relationships with direct suppliers. And this is not a great fit for FOSS because the term direct supplier is not defined anywhere. So the ENISA, EU Cyber Security Agency, clarified that FOSS projects with no contractual relationships are not direct suppliers. And instead of forcing a supplier model, they recommend that operators should support the FOSS community depend on and engage and invest in mutually beneficial relationships.

The fourth example is where policymakers avoid a conflicting regional regime. We saw this happen for the root servers, where the EU decided to not include the root server operators in the NIS2 legislation because it would create overlapping obligations on infrastructure that is essentially global. And we feel that this is a contemporary approach that should be applicable for FOSS too because it's not national movement either. Now, having seen some contemporary approaches, we'll head to the next slide.

We created five actionable guidelines for policymakers. As SSAC, we are indifferent to if you regulate, but if you regulate, please

don't get them wrong. So SAC provides five practical guidelines for creating a FOSS aware policy.

First, acknowledge the critical role of FOSS. Explicitly recognize that critical infrastructure relies on FOSS and this is a strength to be preserved. Consult the FOSS community, engage maintainers, foundations, and operators in policy development to avoid unintended consequences. Guideline 3, make use of the contemporary cases that we just discussed, emerging from the US, UK, and EU. Don't start from scratch. Guideline 4, incentivize FOSS sustainability. Encourage contributions to FOSS as an investment in public infrastructure through grants, tax incentives, and procurement policies. Guideline 5, address systemic risks collectively, fund ecosystem-wide solutions for shared dependencies, security tooling, bug bounties, rather than placing the burden on individual maintainers. Systemic risks, we didn't actually point it specifically out, but it's a risk that is applicable for all software. It's not even FOSS-specific, but also important for FOSS. Now, the common thread here is that policy needs to be designed that fits the FOSS reality, instead of policy that tries to force FOSS in a proprietary model. That's a real tongue twister, right there. Next slide, please.

So, what can ICANN community members do? So for GAC members and policymakers, we suggest you share this with your colleagues working on cybersecurity regulation in your jurisdiction. You can advocate for consulting with FOSS in policy development. You can use the five guidelines to review proposed regulation. Now, the

registry and registrar community can support projects that you depend on financially or with developer time. You can establish internal FOSS policies and share your operational expertise with maintainers.

Now, the technical community can document your FOSS dependencies, contribute back through code, bug reports, or funding, and you can educate others on FOSS operational best practices. For all community members, recognize FOSS as a critical infrastructure that requires investment and help bridge the gap between policymakers and technical operators. The ICANN community is uniquely positioned to influence this issue globally. If this community speaks clearly on the issue and the importance of FOSS, policymakers will listen. Next slide.

We've covered a lot of ground from the technical details of the DNS to the prevalence of FOSS to actionable guidelines for all of us. As we wrap up and move to questions, I want to leave you with the three most important takeaways from this entire presentation.

DNS runs on FOSS. The overwhelming majority of the DNS runs on FOSS. It is a research finding, not an opinion. Policy must start from this fact. FOSS is different, and policy must adapt the FOSS model. Freedom, no contracts has different strengths and risks. Regulation designed for traditional supply chains does not fit and can cause harm. We can get this right. The US, UK, and the EU are developing FOSS aware policies. We have emerging models and five actionable guidelines to follow.

So, a final thought is the security and stability of the DNS depends on understanding and supporting the FOSS ecosystem that underpins it. We would like to thank you. And before we go to questions, I want to credit some specific individuals that made this report possible. We worked on this for about a year, and this would not have been possible without the support from our great staff, in particular, John Emery. We would like to thank the others that were instrumental, including Danielle, and also our great invited guests, including Vittorio, who's right next to me a little bit, and the others that were willing to step into SSAC for the duration of this report. Now, of course, all the work party members who contributed their time and expertise, and my co-chair, Barry Lieba, who guided me through this process as a first-time co-chair.

Now, with credits being distributed, let's head out to questions, if you have any, and thank you for being here today.

DANIELLE RUTHERFORD All righty, we've got a hand up from Andrei Kolesniskov. Andrei, go ahead.

ANDREI KOLESNISKOV Thank you very much. Indeed, it's a great report, and I just want to sharp the image. Everybody knows. Remember this picture where the huge Internet, big blocks, and some little computer somewhere down and on the right side? That's actually attributed to one of the solutions which is dnsmasq. What's the name? Dnsmasq, right?

Which is used in hundreds of millions of the devices worldwide, and it's all on the shoulders of Simon Kelly who is an individual developer for many years developing the software. And this particular case demonstrates the incomparable scale of the dependency of the whole Internet infrastructure, including all our devices—iPhones and cameras and everything, Wi-Fi routers—on the simple little block somewhere down below with the solution. And it's part of the distributives like Debian, Ubuntu, OpenWrt. That's a brilliant example. And thank you very much. It was just my commend.

DANIELLE RUTHERFORD All right. Do we have any other questions from anyone on Zoom or in the room? You do not have to be an SSAC member to ask a question or make a comment.

ENERST MAFUTA KATOKA Greetings, everyone. My name is Enerst Mafuta. I'm an ICANN Fellow. This is new to me and it's very interesting. I would like to say congratulations to the team that put up this together. This is a great research. I just have a question on this FOSS in terms of SAC Advisory, should there be a formal reporting or certification mechanism for this open-source software maintenance in terms of privacy and safety issues?

MAARTEN AERTSEN I think that's not for us to decide. What we are saying here is that if a country, for example, decides to regulate software, either software producers like developers or deployers like critical infrastructure, well, it's their prerogative to do so. What we are saying in this report is if you choose to regulate the space, then you need to be aware that it's mostly FOSS, and if you regulate in a certain way, you run the risk of destroying the good that this model brings. So I know that's a bit of a weasel answer, but yeah, I don't think we, as a committee, should sit on the chairs of policymakers. We are here to look at the technical facts. And this report is perhaps a little bit different in the sense that it is written for a policymaker audience. But still.

ENERST MAFUTA KATOKA Okay. My last question. With the reports that you've written over years, was there any metrics or indication where this advice has been taken by policymakers?

MAARTEN AERTSEN It's a little bit the other way around, to be honest. What inspired me to bring this topic to the SSAC and later become a co-chair for this topic was the policy discussion in Europe. Because Europe, a couple of years ago, decided to start regulating software, and the policy that we can now credit to be innovative and positive to FOSS was in first instance could still use some improvement. So, in that sense, there have been multiple jurisdictions where this has happened already, and I think we can credit the policymakers for

making them FOSS-friendly in the end, but it didn't happen automatically, let's just say that. And we hope that this report will be of value in other jurisdictions where similar discussions will take place in the next decade or so. Maybe there's someone who wants to—yeah, Vittorio?

VITTORIO BERTOLA I just wanted to add on a specific point that, in terms of security, there are already very well-established best practices in terms of how to deal with vulnerabilities when they are reported and disclosure and fixing. So, in the end, in a way, the open source world is already pretty well-established. So there are procedures. There's no need, in a way, for policymakers to introduce new requirements, even if, of course, now at least in certain parts of the world are catching up with the software industry that there are new requirements, but the point is to keep the legal requirements in line with the best practices that have been already established over the decades.

DANIELLE RUTHERFORD All right, thank you. Next in the queue, we have Satish.

SATISH BABU Thanks very much. Once again, thanks for the report. It's fantastic material for us, those of us who are trying to kind of advocate with policymakers on the use of FOSS. I had a question on the difference between governments and businesses on the adoption of FOSS.

For the government, when you don't have a support system, a contractor who will support you, it is pretty hard for them to make a choice in the first place. So we have found we've been kind of lobbying on this from 2001 onwards. We did get our government to insert FOSS into the IT policy. But then they asked us, "Who's going to support this thing? Because no single company is out there." This is 24 years back. So today, also, I what I would like to know is is there a significant difference between CCs and gTLDs in the way FOSS is used? Because I would expect that the governments or CCs will have more difficulties in sticking their necks out and choosing FOSS. But I just wanted to check. Thanks.

MAARTEN AERTSEN Thank you, Satish, for this question. I don't think I have a great answer. What I'm thinking is that you are comparing ccTLD operators to governments. And that is not true in many jurisdictions. In many jurisdictions, the ccTLD operator is actually an entity independent from the government. In this research, for the authoritative DNS services, we did have separate statistics for ccTLDs and gTLDs, but I don't think you can use them to answer your question. So yeah, I don't think we had a look at your question in this report. I do like the question. And I know that others are actually have been advocating for this, like you, for a long time. And it is an actual problem that developing software is not the same as supporting deployment. So yeah, I hope I could still offer some thoughts.

DANIELLE RUTHERFORD All right, go ahead.

JAMES GANNON James Gannon. I run product at an open-source Foundation.

DANIELLE RUTHERFORD Could you speak a little bit closer? You really do have to get very close to the microphone.

JAMES GANNON These mics are really bad. James Gannon is my name. I run product at an open-source Foundation, not in the DNS space.

With NIS2, we came very close to a very bad outcome, and it took a lot of last-minute input, and let's call it lobbying from actors around this table and others. Do you think that looking forward, we have this as the start, it's an SSAC Report, do you think SSAC will do more monitoring going forward of things that may potentially impact the FOSS DNS ecosystem? Or is this report something that could be taken maybe by other people outside of SSAC that goes into a more monitoring? I'd be interested. How do we avoid what we ended up with NIS2 where it came in really at the last minute, and there was a big concerted effort, rather than it being a ongoing monitoring and education with policymakers? Because certainly, from my day job, I'm in healthcare space, policymaker interactions are long-term things that you do slowly. It looks not great if you

come in at the end and have to kind of go, “Whoa, whoa, whoa, whoa.” So is there something we can pivot from this report within the ICANN ecosystem?

MAARTEN AERTSEN It’s a very good point. I’m afraid that SSAC is not really set up to monitor global policy development. We could even say that we’re better set up to monitor technical developments. And this is kind of out of our normal comfort zone maybe. So, this morning, we gave a presentation to ICANN Board, and one of the reflections we had at the end is that ICANN’s government relations team might be in a very good position to spot policy developments in various jurisdictions, and then surface this research, and maybe also prompt us to do new research as needed.

So, other than that answer, I think you made a few really good points about how interaction with policymakers work. I don’t think it’s something for SSAC, but I definitely hope that if you are here today or if you watch this recording in the future, you’ll take the report to the people who actually make the decisions, because only then will we have created value. Otherwise, we will just have created some more paper that sits in some desk or on some hard disk. Thank you.

DANIELLE RUTHERFORD Georgia, go ahead.

GEORGIA OSBORN Hi. Not a question but a comment. I'm also piggybacking off the gentleman's comment over there. I think that the presentation that you've just done is going to be incredibly useful for regulators going forward. It's one thing to show them a very long report, which may be very complex, but another thing to bring a presentation like this, which is excellent, and to kind of get them to see the value of it and explain it in the way that you've just done. So I really appreciate it and I completely agree with the gentleman's point over there. It is going to be a process, but yeah, I guess that's not for SSAC to do, but I appreciate the presentation a lot for that reason. Thank you.

MAARTEN AERTSEN Thank you, Georgia. Then I hope we smiled enough at the camera while speaking.

DANIELLE RUTHERFORD All right, looking around, I'm not seeing hands. Any other questions or comments for Maarten and Tara? Going once, going twice.

MAARTEN AERTSEN In that case, I would like to thank you all for attending today. If you have any questions later on, be sure to just talk to us in the hallway.

DANIELLE RUTHERFORD Thank you so much for joining us today. Please stop the recording.

[END OF TRANSCRIPTION]