SAC132
The Domain Name System Runs on Free and Open Source Software (FOSS)

A Report from the ICANN Security and Stability Advisory Committee (SSAC)
25 September 2025

Preface

This is a report to the ICANN Board, the ICANN organization, the ICANN community, and, more broadly, the Internet community from the ICANN Security and Stability Advisory Committee (SSAC) about the ways in which the Domain Name System (DNS) is dependent upon Free and Open Source Software (FOSS).

The SSAC focuses on matters relating to the security and integrity of the Internet's naming and address allocation systems. This includes operational matters (e.g., pertaining to the correct and reliable operation of the root zone publication system), technical administration matters (e.g., pertaining to address allocation and Internet number assignment), and registration matters (e.g., pertaining to registry and registrar services). SSAC engages in ongoing threat assessment and risk analysis of the Internet naming and address allocation services to assess where the principal threats to stability and security lie, and advises the ICANN community accordingly. The SSAC has no authority to regulate, enforce, or adjudicate. Those functions belong to other parties, and the advice offered here should be evaluated on its merits. SSAC members participate as individuals, not as representatives of their employers or other organizations. SSAC consensus on a document occurs when the listed authors agree on the content and recommendations with no final objections from the remainder of the SSAC, with the exception of any withdrawals included at the end of the document.

Table of Contents

List of Figures

List of Tables

Executive Summary

The Domain Name System (DNS) is a globally distributed, hierarchical, and decentralized system whose information underpins nearly every online interaction. Its principal purpose is to map user-friendly domain names to the computer-friendly IP addresses required to locate resources on the network. Whether browsing the web, sending an email, or using a mobile application, every online connection relies on information that originates from and is structured by the DNS.

This report's core finding is that the DNS is built and sustained on Free and Open Source Software (FOSS). This is not a niche practice, but the dominant reality. FOSS is the norm for the most fundamental components of DNS infrastructure. For example, at least nine of the 12 independent organizations that operate the Internet's root server system (RSS) exclusively use FOSS implementations, and nine of the 10 largest service providers for top-level domains (TLDs) use FOSS. This dominance stems from the inherent strengths of the FOSS development model, which combines economic efficiency and low-friction adoption with the transparency, collaborative security, and operational resilience essential for critical infrastructure.

Although the FOSS development model is fundamentally different from that of proprietary software, FOSS is not inherently more or less secure. The security of any software project is determined by the quality of its development and maintenance processes, not the visibility of its source code. Unlike commercial software, FOSS is a collaborative, global effort built upon four essential freedoms: to use, study, share, and change. This ecosystem depends on a global network of maintainers and contributors who are often unpaid volunteers. While many are unpaid volunteers, the DNS space is unique in also relying on a handful of long-lived maintenance organizations. This creates a model based on community collaboration rather than the commercial contracts that define a traditional software supply chain, which introduces unique risks related to financial sustainability for the maintenance organizations and maintainer burnout for volunteers.

These unique characteristics mean that regulatory frameworks designed for proprietary software may not be well-suited for FOSS and therefore could have severe unintended consequences to the stability of critical Internet infrastructure. To navigate these complexities and foster a secure digital ecosystem, the Security and Stability Advisory Committee (SSAC) provides the following guidelines for policymakers:

- **Acknowledge the Critical Role of FOSS:** Policymakers should explicitly acknowledge in any relevant legislation or regulation that critical Internet infrastructure relies on FOSS, and that its use is a strength to be preserved.
- **Consult the FOSS Community**: Developing legislation and regulations should be informed by consulting all parts of the FOSS ecosystem, from individual maintainers to non-profits and corporations.

- **Make Use of the Contemporary Cases in FOSS Regulation:** Policymakers can reference recent case studies in the report of contemporary approaches that incorporate the unique characteristics of the FOSS development model.
- **Incentivize FOSS Sustainability:** Encourage public and private sector contributions to critical FOSS projects as a form of investment in a shared public good.
- **Address Systemic Risks Collectively:** Foster and fund collaborative, ecosystem-wide solutions to mitigate risks from shared dependencies instead of burdening individual maintainers.

1 Introduction

This report is motivated by the increased involvement of policymakers in industry efforts to decrease software vulnerabilities in digital infrastructure. As governments and regulatory bodies around the world seek to secure the software supply chain, it is critical that these efforts are informed by a clear understanding of how the Internet's most foundational systems are actually built and maintained. Recent examples of (proposed) regulatory intervention to decrease software vulnerability in digital infrastructure include:

- A voluntary software security code of practice for software vendors in the UK.[1]
- Industry self-attestation of secure software development practices for use by the US government.[2]
- Market entry requirements ("Cyber Resilience Act (CRA)") for digital products (including software) in the EU.[3]
- Risk management and reporting obligations for digital infrastructure providers in the EU ("NIS2 IA").[4]
- Open source communities are named as a development objective in China's five-year plan for IT.[5,6]

Although Free and Open Source Software (FOSS) is the dominant software development practice today, its particular characteristics are often overlooked in policy discourse. If policymakers introduce regulatory interventions without understanding the unique development and supply model of FOSS, they risk threatening the security and stability of the critical infrastructure that depends on it, including the Internet's domain and routing systems. This report aims to provide that necessary context.

Contrary to what is usual in other sectors, much of the software that runs the Internet is available under FOSS copyright licenses. These licenses are not primarily about cost; they are about freedom. Specifically, FOSS licenses grant operators of infrastructure four essential freedoms: to use, study, change, and share the software—modified or not—with everyone. This is a

[1] "Software Security Code of Practice." UK Department for Science, Innovation & Technology, 7 May 2025. https://www.gov.uk/government/publications/software-security-code-of-practice/software-security-code-of-practice.
[2] "Secure Software Development Attestation Form." US Cybersecurity and Infrastructure Security Agency, https://www.cisa.gov/secure-software-attestation-form.
[3] "Cyber Resilience Act: MEPs adopt plans to boost security of digital products," European Parliament press release, March 12, 2024, https://www.europarl.europa.eu/news/en/press-room/20240308IPR18991/cyber-resilience-act-meps-adopt-plans-to-boost-security-of-digital-products
[4] "Cyber Resilience Act: MEPs Adopt Plans to Boost Security of Digital Products." Press Release. European Parliament, 12 March 2024. https://www.europarl.europa.eu/news/en/press-room/20240308IPR18991/cyber-resilience-act-meps-adopt-plans-to-boost-security-of-digital-products.
[5] "14th Five-Year Plan for the Software and Information Technology Services Industry." State Council of the People's Republic of China, Ministry of Industry and Information Technology, December 2021. https://www.gov.cn/zhengce/zhengceku/2021-12/01/5655205/files/a44b507d67c74591ad4f5e55b98c4518.pdf.
[6] "Translation: 14th Five-Year Plan for National Informatization." DigiChina Project, Stanford University, 24 January 2022. https://digichina.stanford.edu/work/translation-14th-five-year-plan-for-national-informatization-dec-2021/.

development model, not just "free-of-charge" software, [7] and it is the foundation of the collaborative, global effort that builds and maintains much of the Internet's critical infrastructure.

This report is structured to provide policymakers with a comprehensive understanding of the role of FOSS in the Domain Name System (DNS) and domain name registration ecosystem.

- Section 2 provides a non-technical DNS Primer, explaining the key components and functions of this critical infrastructure.
- Section 3 details the FOSS Model, explaining its key characteristics, inherent strengths, and unique risks compared to proprietary software.
- Section 4 presents the Security and Stability Advisory Committee's (SSAC) core research on the prevalence of FOSS, demonstrating its dominance in the most critical parts of the DNS.
- Section 5 examines several contemporary cases from the United States, United Kingdom, and European Union that illustrate how policymakers are adapting cybersecurity regulations to the unique realities of the FOSS ecosystem.
- Section 6 consolidates the report's core analysis into a series of findings that form the evidentiary basis for the guidelines that follow in Section 7.
- Section 7 provides direct, actionable guidelines for policymakers.

2 DNS Primer

When you send an email, browse a webpage, chat with friends, etc., your device (e.g., computer, phone, or tablet) is sending and receiving thousands of pieces of information. You can think of this like a digital postcard, with a sender address, a receiver address, and content. Every device connected to the Internet has at least a unique Internet Protocol (IP) address associated with it. For humans, it is easier to remember names rather than numbers. The DNS is like the address book of the Internet. It connects the IP address to the domain name so that everyone can more easily navigate the Internet.[8,9] The DNS is a critical system that is vital to the continued stable, secure, and interoperable global Internet.

[7] FOSS can be used for any purpose and is free of restrictions such as license expiry or geographic limitations. Its code can be studied by anyone, without non-disclosure agreements or similar restrictions. It can be shared and copied at virtually no cost. Additionally, FOSS can be modified by anyone, and these improvements can be shared publicly. The absence or weakening of at least one of these freedoms means an application is proprietary, so non-open source software. The four freedoms are given by a software license. Software licenses define the conditions under which a program can be used and reused. For it to be Free Software, the license text must contain at least the four freedoms. The Free Software Foundation (https://www.gnu.org/licenses/license-list.html), the Open Source Initiative (https://opensource.org/licenses), and the Debian Project (https://wiki.debian.org/DFSGLicenses) maintain lists of reviewed and approved licenses. An application can usually not be considered FOSS if its license does not appear in one of these lists.

[8] ICANN. "The Domain Name System," 13 September 2022. https://www.icann.org/resources/pages/dns-2022-09-13-en.

[9] Cloudflare. "What Is DNS? | How DNS Works," https://www.cloudflare.com/learning/dns/what-is-dns/.

The DNS provides a mapping from user-friendly domain names (e.g., icann.org) to computer-friendly numerical IP addresses (e.g., 192.0.43.7 or 2001:db8::1). These mappings collectively make up a global namespace. In order to have a website or email account that is accessible via a domain name, a domain holder must publish the mappings for that service. This publication in the DNS makes the connection between the domain name and the service's IP address available to anyone on the Internet.

As Figure 1 illustrates, the DNS ecosystem has three main parts:

- Domain Name Registration (light blue box): This is the administrative and contractual framework for acquiring domain names.
- DNS Infrastructure (green box): These are the technical systems that make domain names work on the Internet by translating them into IP addresses.
- End User and Content Services (orange and dark blue boxes): These are the services that people ultimately use, such as websites and email.

The focus of this report is the domain name registration and DNS infrastructure, which together form a layer of the Internet's critical infrastructure. Information lookup in the global domain namespace is mostly not monetized and provided "free," without charge, to clients. The infrastructure that enables this global, distributed service as a "non-exploitative public good" is what this report refers to as "DNS infrastructure." The system is widely distributed across the Internet consisting of multiple components, operated by many independent organizations playing different specific roles in this system. These systems interoperate because the necessary technical details of their interfaces and communications are standardized through the Internet Engineering Task Force (IETF).

This critical infrastructure is separate from the content that is delivered over it. Registering a domain name is not the same as creating a website. Publishing a domain name in the DNS so it can be found is not the same as publishing website content. This report examines the foundational software that makes the address book work, not the information written on the pages of that book.

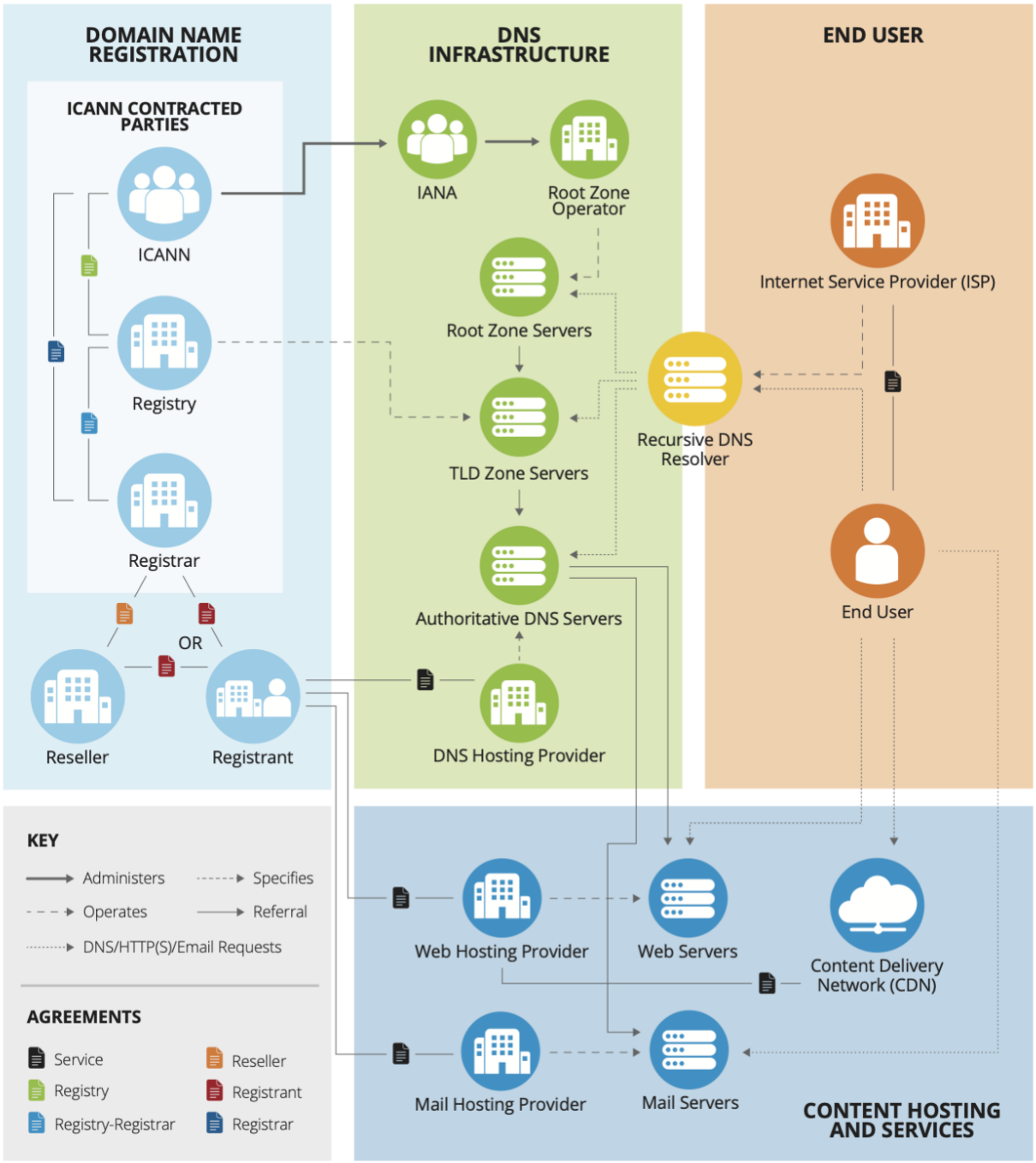

Figure 1: *The Internet Ecosystem.* Drazek, Keith. "DNS Security: Ongoing Community Work to Mitigate Domain Name System (DNS) Security Threats." Verisign Blog, 7 December 2021. https://blog.verisign.com/domain-names/ongoing-community-work-to-mitigate-domain-name-system-security-threats/.

2.1 DNS Hierarchy

The DNS is a globally distributed, hierarchical, and decentralized system. This system, shown as the "DNS Infrastructure" in Figure 1, is designed for resilience, ensuring there is no single point of failure for the global Internet.

Figure 2 shows how the DNS namespace is hierarchical.

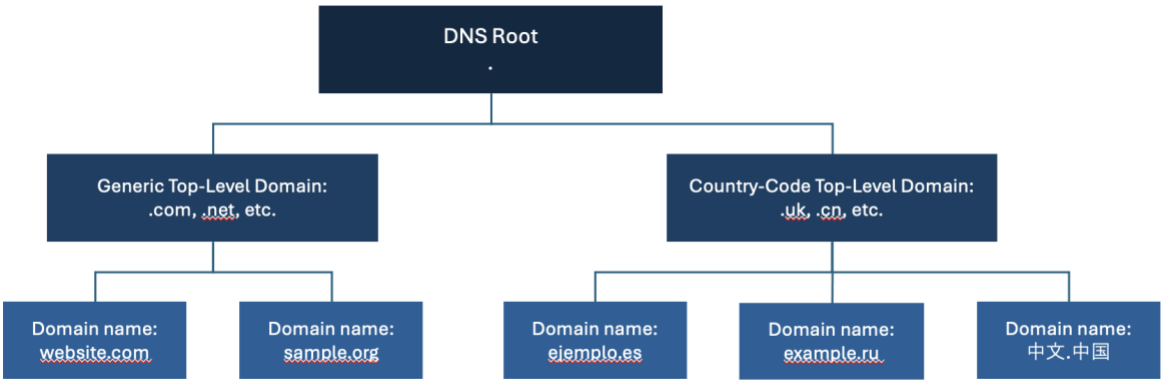

Figure 2: *DNS Hierarchy*

At the very top of the hierarchy is the DNS Root (.), which is served by the root server system (RSS).[10] The root points to the authoritative servers for various Top-Level Domains (TLDs). TLDs are the endings of domain names and are broadly categorized into two types:

- Generic TLDs (gTLDs): These are general-purpose domains like .com, .org, .xyz, and .shop.[11]
- Country-code TLDs (ccTLDs): These are reserved for use by countries, territories, and geographical locations identified in the ISO 3166-1 Country Codes list.[12] ccTLDs can base their names on the two-letter country codes defined by the ISO 3166-1 standard (e.g., .jp for Japan, .fr for France, .ke for Kenya), or they can represent a country or territory name in a script other than ASCII characters.[13] This is known as Internationalized Domain Names (IDNs), a concept that has been implemented by ICANN since 2009.

[10] The root server system is comprised of globally dispersed servers operated by 12 independent organizations. For more information on the root server operators, see "RSSAC023v2: History of the Root Server System." ICANN Root Server System Advisory Committee (RSSAC), 17 June 2020. https://itp.cdn.icann.org/en/files/root-server-system-advisory-committee-rssac-publications/rssac-023-17jun20-en.pdf.

[11] ICANN. "Acronyms and Terms, Generic Top-Level Domain (gTLD)," https://www.icann.org/en/icann-acronyms-and-terms/generic-top-level-domain-en.

[12] International Organization for Standardization. "ISO 3166 — Country Codes," https://www.iso.org/iso-3166-country-codes.html.

[13] ICANN. "Acronyms and Terms, Country Code Top-Level Domain (ccTLD)," https://www.icann.org/en/icann-acronyms-and-terms/country-code-top-level-domain-en.

Below the TLDs are the individual domain names. A domain name consists of two or more textual segments separated by dots. As shown in Figure 3, a domain name is constructed from right to left, starting with the TLD. For example, in icann.org, the TLD is .org and the second-level domain is icann. Together, they form the unique domain name. In the second example, bbc.co.uk, the TLD is .uk and the second-level domain is .co. This hierarchical and decentralized system allows for a robust and resilient global Internet.

A domain name is a unique name that forms the basis of the uniform resource locators (URLs) that people use to find resources on the Internet. The domain name itself identifies a specific address on the Internet that belongs to an entity such as a company, organization, institution, or individual.[14]

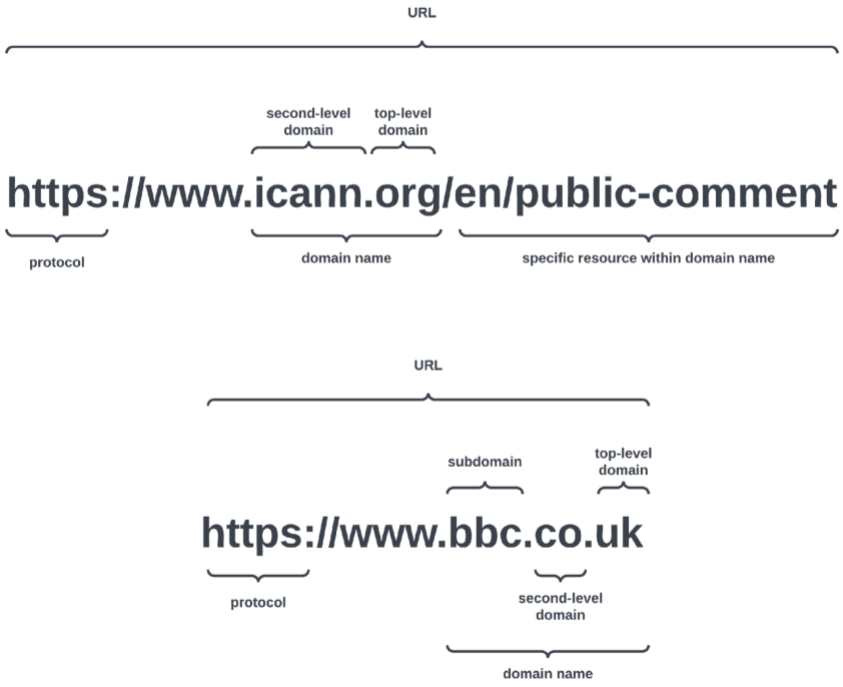

Figure 3: *Components of a URL and Domain Name*

2.2 Domain Name Registration and Publication

The domain name registration infrastructure refers to the systems that facilitate the acquisition of a unique domain name. This process, illustrated in the "Domain Name Registration" portion of Figure 1, involves several key actors and two distinct processes: registration and publication.

Registration is the process of reserving a domain name. A registrant is the individual or organization that registers and holds the rights to a specific domain name. To register a domain name under a specific TLD, the registrant must use a registrar. The registrar is a public-facing organization that acts as a retailer for domain names. Registrars are effectively the distribution

[14] ICANN. "Acronyms and Terms, Domain Name," https://www.icann.org/en/icann-acronyms-and-terms/domain-name-en.

channel for registrations, managing payments, renewals and other administrative information. The registrar then interacts with a registry. The registry is the authoritative, master database of all domain names within that TLD. The organization that maintains this database is called the registry operator. To automate the millions of transactions between them, registrars and registries use the Extensible Provisioning Protocol (EPP), a standardized protocol for domain name registrations, renewals, and transfers.

The outcome of a successful registration is the publication of the domain's DNS records on the authoritative servers that are part of the DNS infrastructure, which makes the domain reachable on the Internet. Publication is the technical process of making a domain's DNS records available on authoritative nameservers. These servers hold the listings of user-friendly domain names to their corresponding computer-friendly IP addresses, which is critical for locating and verifying the authenticity of websites, mail servers, and other Internet resources.

2.3 DNS Resolution

When you type a domain name like www.example.com into a web browser, a process called "DNS resolution" begins. This process relies on two types of DNS infrastructure components that work together: recursive resolvers and authoritative servers (see Figure 4).

Figure 4: *Traditional DNS Resolution*

Authoritative Servers are the components that publish domain information. They hold the definitive, official record for a specific domain and make it available for retrieval. Authoritative nameservers are operated by individuals, enterprises, universities, service providers, and government entities. TLD nameservers, for example, publish the technical data for all the individual domains within their registry. Many organizations have local DNS services that only provide information that applies to their internal organization, such as for locating departmental websites on an intranet. This is a large part of the DNS that is not visible from the open Internet.

Some end-users and small businesses rely on their domain name registrar or webhoster's authoritative services to publish their domain information, but it is common for companies, governments, and organizations to run their authoritative servers separately, either in-house or outsourced. Key concerns are reliability and redundancy; as such, administrators of the "zone" information published under a domain often purchase "secondary authoritative DNS services"

from external providers. Those providers publish the same data as the primary authoritative DNS servers. Many of the organizations offering these services are the same that deliver authoritative DNS services for TLDs.

Resolvers facilitate the retrieval of DNS information. They act on behalf of a user's device (a "client," such as your smartphone) to find the correct IP address. To do this efficiently, resolvers maintain a constantly-refreshed database of recent lookups, called a cache. In practice, it is common for resolvers to answer up to 90% of queries from their local cache, which is much faster than querying servers across the Internet. For answers they do not have cached, they must query authoritative servers. Resolvers are typically very busy, heavily loaded systems, in part because a consistent high traffic level ensures a constantly-refreshed cache.

There are millions of resolvers around the world. The resolver function can be local, within the user's network or the access provider's network, or in the cloud, either as a separate service hosted on the Internet or in conjunction with other cloud services. Regardless of how the service is set up, it is important to ensure that only intended users are able to query the service. If the service is open to anyone on the Internet, it could be abused, e.g., Distributed Denial of Service (DDoS) attacks. Configuration in the user's device typically determines the preferred provider. Many universities, enterprises, and Internet service providers (ISPs) provide local resolvers for users on their networks, and there are popular cloud services such as Google (8.8.8.8), Quad9 (9.9.9.9), and Cloudflare (1.1.1.1). Key factors in deciding which resolver provider to use include organizational policy, the filtering or blocklist policies of the resolver, and availability of encrypted DNS transports. The resolution process (Figure 4) shows how these components work together following the DNS hierarchy.

3 The FOSS Model: Key Characteristics and Implications

The previous section provided a primer on the DNS and domain name registration infrastructure, the foundational systems that allow users to navigate the Internet. This section will explain how that critical infrastructure is built and maintained. The software that underpins much of this infrastructure is FOSS, which operates on a fundamentally different development and economic model than traditional proprietary software. An informed policy discussion requires an understanding of these unique characteristics

3.1 Key Roles in the FOSS Ecosystem

Unlike traditional closed-source software where development is handled internally by a single legal entity, the FOSS model is open and distributed. The roles are not mutually exclusive and can be filled by anyone, anywhere. For the purposes of this report, the following terms will be used:

- **Maintainer:** The individual or group responsible for the overall direction of a FOSS project. Maintainers have the authority to accept or reject contributions to the official

version of the software, ensuring quality and consistency. They are the stewards of the project.

- **Contributor:** An individual or organization that offers improvements to a project, such as by submitting code, documentation, or bug reports. These contributions are reviewed by a maintainer before being included in the official project.
- **User or Operator:** The individual or organization that deploys and uses the software. In the context of the DNS, an "operator" is an entity that runs DNS infrastructure components, such as authoritative servers or resolvers. Users and operators are a vital part of the ecosystem, providing feedback and giving the project a purpose.

To avoid ambiguity, this report uses the specific terms "maintainer" and "contributor" rather than the more general term "developer," as an individual might develop code as a maintainer, a contributor, or independently.

3.2 Core Principles of FOSS Development Model

The FOSS model is defined by the rights granted by its copyright licenses. For a program to be considered FOSS, its license must grant users four essential freedoms:[15,16]

- The freedom to ***use*** the software for any purpose, without restrictions such as license expiry or geographic limitations.
- The freedom to ***study*** how the program works, which requires access to the source code without non-disclosure agreements.
- The freedom to ***share*** the software, meaning it can be redistributed and copied at virtually no cost.
- The freedom to ***change*** the program and release those improvements to the public.

The absence or weakening of any one of these freedoms means the software is proprietary, not FOSS. These freedoms are the foundation for the general characteristics of FOSS, which differ significantly from proprietary software. Organizations such as the Free Software Foundation,[17] the Open Source Initiative,[18] and the Debian Project[19] maintain lists of software licenses that have been reviewed and approved as meeting these criteria.

These four freedoms are not just abstract principles; they are the foundation for the unique development model and economic characteristics of FOSS, which are described in the following subsections.

---

[15] GNU Operating System. “What Is Free Software?,” https://www.gnu.org/philosophy/free-sw.html.
[16] An alternative and popular definition of what constitutes a FOSS license is Open Source Initiative’s “Open Source Definition,” https://opensource.org/osd.
[17] GNU Operating System. “Various Licenses and Comments about Them,” https://www.gnu.org/licenses/license-list.html.
[18] Open Source Initiative. “OSI Approved Licenses,” https://opensource.org/licenses.
[19] Debian Wiki. “DFSGLicenses,” https://wiki.debian.org/DFSGLicenses.

3.2.1 The FOSS Model Allows for Global Collaborative Development

Because FOSS licenses grant anyone the freedom to study, modify, and share the code, they facilitate open global collaboration in software development. There is no arbitrary set of preconditions that is applied to contributors to FOSS development, permitting any individual to make contributions to the software. Contributions can come from a diverse array of individuals and organizations, sometimes as part of a corporate development of a product, sometimes on a voluntary individual basis.

The resulting product is made freely available to anyone not just for any use (including commercial), but for further development. This reduces friction in the evolution of software, in particular where modifications are integrated back into the software product, contributing to and speeding up feature development and the resolution of software bugs and security issues. The collaborative nature of FOSS fosters transparency and rapid innovation. Moreover, contributions themselves are normally also open, allowing others to inspect proposed contributions and comment as to their suitability of the contribution and the merits of integration of the contribution into the software product.

However, it is uncommon for all such contributions to be accepted without some level of inspection and acceptance. Such checks are intended to decrease the risk of software compromise through the inclusion of inconsistent, incorrect or rogue code. There are some common conditions to the acceptance of contributions, including the need to preserve the copyright and intellectual property considerations. A common FOSS practice is that all collaborative contributions cannot impose additional constraints on the use of the software beyond the conditions already associated with the software, nor can they impose additional copyright or intellectual property constraints on the existing software base.

As an alternative to contributing, FOSS licenses also enable contributors to start their own derivative (fork), instead of working with the original project (upstream). Permissionless forking enables rapid innovation. Some forks have become more popular than the upstream they derive from. But they can also create confusion, splitting available review capacity to spot errors or malicious additions.

Software can be self-contained, or it can rely on other software libraries to construct the necessary functionality. Such libraries, essentially external code components, can change in ways that are less visible to the code's developers and this situation can produce unintended behaviors. FOSS has a very low barrier to entry for use as a component in other (including proprietary) software, which makes the observed frequency of occurrence higher for such risk, even though it is not unique to FOSS.

3.2.2 The FOSS Model Functions Without Contractual Relationships

One major distinction of the FOSS supply model is that these parties usually do not have a contractual relationship. This contrasts with the supply chain for physical goods, where manufacturers and distributors have contracts that can be leveraged to attach policy or transfer

compliance obligations. However, in the FOSS supply model, operators obtain software through multiple pathways. While a maintainer provides the software in source form for public download, it is more commonly distributed through intermediaries that provide installable packages (e.g., the "Debian project") or products or services. Beyond the terms of the maintainer's open source license, an operator rarely has a contract with such an intermediary. Only a fraction contract with maintainers. When a contractual relationship exists it is usually for technical support, not for the FOSS itself.

3.2.3 The FOSS Model Decouples Funding From Use

Unlike physical goods, software is mere information and does not have an inherent material cost per additional unit produced. The freedom to use and share FOSS means that licenses grant rights without requiring an exchange of money. Users, including operators of Internet infrastructure, can fund FOSS development and maintenance if they want, but the license does not require them to, so funding for developers is not tied to the use of their software.

3.2.4 FOSS often has no single responsible legal entity

In the world of physical goods, it is quite common to assume the existence of a single responsible legal entity, which in turn can be made subject of a policy, or carry a compliance burden. Unlike physical goods, distributing software globally often happens at very low or zero cost over the Internet. This enables individuals, or groups of individuals, to write and distribute software without payment and without creating a legal entity. The Debian project[20] is a high-profile example of an open source project without an associated legal entity. Debian software is used in such fields as nuclear energy, rail transport, industrial automation, and healthcare equipment. While some FOSS projects have a legal entity providing governance, sponsorship or even employment, most do not. We will see later that DNS is unusual in this regard, with four small organizations professionally maintaining FOSS.

3.3 Proprietary Systems Depend on FOSS

The characteristics of FOSS turn out to be relevant to systems that themselves are proprietary (not FOSS). This is due to the fact that modern proprietary systems often rely on FOSS to function. FOSS tools and libraries are ubiquitous in proprietary software development and deployment. For example, most software is not produced without compilers or run-time environments; software cannot store objects or structured data without object stores or databases; software cannot communicate with other components without a facility for message passing. These back-end functions are commonly provided by FOSS.

In the specific case of the DNS, Cloudflare provides a useful example. Cloudflare operates a widely-used public DNS resolver service which is known as "1.1.1.1". The service was originally provided successfully using Knot Resolver, which is FOSS; however it was later

[20] Debian. "Our Philosophy: Why We Do It and How We Do It," https://www.debian.org/intro/philosophy.

replaced by proprietary software that was developed internally.[21] The proprietary software that runs at the heart of 1.1.1.1 is written in Rust, a programming language whose specification, reference implementation and compiler is provided as FOSS by a not-for-profit foundation.[22] It makes use of tokio,[23] an asynchronous runtime that is also FOSS. It runs on Linux, a FOSS operating system, and is surrounded by a considerable array of other FOSS components that provide observability and other operational necessities.[24] Cloudflare's proprietary software behind 1.1.1.1 operates at scale thanks to FOSS, not in spite of it.

3.4 Inherent Strengths of FOSS in the DNS Ecosystem

The FOSS model is not just a theoretical construct; its principles translate into tangible benefits that have made it the dominant paradigm for DNS infrastructure software. The freedoms to study, share, and improve software collectively create an environment that fosters transparency, resilience, and innovation. These are not accidental byproducts, but rather inherent strengths of the FOSS model that have proven to be particularly well-suited to the demands of building and maintaining critical Internet infrastructure. The following sections detail these specific strengths.

3.4.1 Transparency and Collaborative Security

Open source DNS implementations benefit from transparency. The availability of source code for FOSS implementations has enabled a global community of developers, researchers, and operators to identify and address vulnerabilities, often more quickly than in proprietary systems. They have been an active subject of study by the academic and security communities for decades,[25] leading to vulnerability reporting both in the DNS protocol and in the open source implementations.[26]

If vulnerabilities are identified, the close ties between the three groups allow for the timely addressing of any serious issue and for the coordination of disclosures and patch releases, ensuring that critical DNS servers (such as the root ones) are fully secured before the information is made public, and that all other servers are promptly updated.

In a survey among operators of DNS infrastructure (see Appendix C), respondents indicated they valued visibility into their supply chain transparency as a powerful characteristic that allows

[21] Wen, Anbang, and Marek Vavruša. "How Rust and Wasm Power Cloudflare"s 1.1.1.1." The Cloudflare Blog, 28 February 2023. https://blog.cloudflare.com/big-pineapple-intro/.
[22] The Rust Foundation. "About Us - Mission, Leadership, Board," https://rustfoundation.org/about/.
[23] Tokio. "Tutorial," https://tokio.rs/tokio/tutorial.
[24] Graham-Cumming, John. "CloudFlare And Open Source Software: A Two-Way Street." The Cloudflare Blog, 7 October 2013. https://blog.cloudflare.com/open source-two-way-street/.
[25] For example see Paul Vixie's paper, "DNS and BIND Security Issues," in *Proceedings of the Fifth USENIX UNIX Security Symposium*, Salt Lake City, UT, 1995. https://www.usenix.org/legacy/publications/library/proceedings/security95/full_papers/vixie.pdf.
[26] To illustrate the notion "active subject of study," see, for example, the summary of findings from 23 research papers published in 2024 on DNS security, as presented by Chaoyi Lu, "Summarizing DNS & Security Academic Papers Published in 2024" (presentation, ICANN82, Seattle, WA, 12 March 2025), https://static.sched.com/hosted_files/icann82/7b/1.1%20chaoyi-dnspapers2024-0304.pdf.

them to expedite patching of vulnerabilities in software dependencies of the DNS service they operate. For example, see the following responses:[27]

> Open source QUICKLY exposes any flaws -- whether security-related or otherwise -- getting patched and patches distributed far faster than any commercial entity.
>
> Large commercial entities such as [redacted], [redacted], and [redacted] have often over the course of years released security patches far AFTER the fact of the vulnerability's discovery.
>
> We always advocate that using FOSS is in the interest of everyone, both users and consumers of our services. It allows them to verify that we are using sound software components. In the rare cases where a security problem is detected in the software we use, it is openly discussed and fixed at top speed -- quite unlike what we see with proprietary software.
>
> There are plenty of examples of proprietary software with security problems and poor track records of transparency or fixes. The FOSS community has done a better job than most vendors in both of these.

The use and management of FOSS DNS software is actively and overtly discussed by operators in public. Each of the leading DNS software systems has an open public user mailing list,[28] and a world-readable public bug or issue database.[29] The DNS community collaborates on operations and research via the industry organization, DNS Operations, Analysis, and Research Center (DNS-OARC).[30] This culture of open communication contributes to faster and wider awareness of operational problems, software bugs, incompatibilities between implementations, and operator errors that impact other parts of the system.

3.4.2 Stability and Long-Term Support

FOSS in general has a widespread reliance on volunteer maintenance, often by a single person. In the case of DNS, four leading open source solutions are made by four different organizations (three non-profit institutions and one commercial company) from multiple countries and territories in North America and Europe.[31] All four were started in the early days of mass Internet adoption and have achieved stability in their sources of revenue and internal

[27] ICANN Security and Stability Advisory Committee, "ICANN SSAC Survey on the Anticipated Impacts of Open Source Regulation on the DNS Infrastructure," personal communication, February 2025.

[28] For example, see public mailing lists by CZnic (https://lists.nic.cz/postorius/lists/knot-resolver-users.lists.nic.cz/), NLnet Labs (https://www.nlnetlabs.nl/support/mailing-lists/), and PowerDNS (https://mailman.powerdns.com/mailman/listinfo/).

[29] For example, see issue tracking for CoreDNS (https://github.com/coredns/coredns/issues) and BIND 9 (https://gitlab.isc.org/isc-projects/bind9/-/issues/).

[30] DNS Operations, Analysis, and Research Center. "Introduction to DNS-OARC," 3 July 2008. https://www.dns-oarc.net/oarc/info.

[31] In alphabetical order: CZNIC, an association of Czech Internet Service Providers ("ISPs") founded in 1998; Internet Systems Consortium, Inc. ("ISC"), a US non-profit corporation established in 1994 expressly to support open source software and systems for Internet infrastructure; NLnet Labs, a Dutch non-profit founded in 1999 to develop open standards and open source software for DNS and inter-domain routing; and PowerDNS, a Dutch company founded in 1999 to support the development of specialized DNS software for ISPs.

organization. Each of these organizations leads the development of their software by writing new code and by reviewing and vetting code contributions from active communities scattered around the globe. The long-term commitment of these sponsoring organizations and the centralized technical control of software changes help to mitigate the quality, security, and reliability concerns common to FOSS projects developed without strong technical controls. The culture of the open source DNS technical community is such that it would require a long duration sustained effort for an infiltrator to build up the relationships and reputation required for a trusted role.

3.4.3 Operational Resilience Through Diversity

The high availability required of DNS services implies the need for redundancies and avoidance of single points of failure. For many operators, this includes operating multiple independent implementations of DNS in software. Operators who outsource can similarly pursue a multi-provider strategy. This creates intrinsic demand for the existence of several solutions, so that operators of highly reliable services can run two or more in parallel and avoid being dependent on any individual software developer or entirely vulnerable to problems in any single product. The availability of FOSS DNS software lowers the barrier to organizations who want to operate their own infrastructure, providing a way out of market concentration and centralization.[32]

3.4.4 Enabler of Economic Growth and Innovation

FOSS is known to be an enabler of growth.[33,34] As we saw, multiple open source products are readily and freely available to anyone needing DNS software, anywhere in the world. Likely users are ISPs, domain name registries and registrars, and Internet users. Individuals, non-profits, startups, and subject matter experts that might not be able to afford the cost of proprietary software licenses could get the software they needed for free. As a respondent to a survey of DNS operators stated (see Appendix C): "My biggest concern today is that we use so many open source components in noncritical parts of our stack, that we do not have support contracts for, that in totality we could not afford licensing even if it were available."[35]

The existence of multiple FOSS DNS solutions has a positive effect on the technical and commercial development of Internet usage and of online software and services in general, both open source and proprietary. Many proprietary systems and hosted services rely heavily on open

[32] Nottingham, Mark. "RFC 9518: Centralization, Decentralization, and Internet Standards." Request for Comments. Internet Engineering Task Force, 18 December 2023. https://datatracker.ietf.org/doc/rfc9518/.
[33] "The analysis estimates a cost-benefit ratio of above 1:4 and predicts that an increase of 10% of OSS contributions would annually generate an additional 0.4% to 0.6% GDP as well as more than 600 additional ICT start-ups in the EU. Case studies reveal that by procuring OSS instead of proprietary software, the public sector could reduce the total cost of ownership, avoid vendor lock-in and thus increase its digital autonomy." From the European Commission. Directorate General for Communications Networks, Content and Technology. *The Impact of Open Source Software and Hardware on Technological Independence, Competitiveness and Innovation in the EU Economy: Final Study Report*. LU: Publications Office, 2021. https://data.europa.eu/doi/10.2759/430161.
[34] Wright, Nataliya Langburd, Frank Nagle, and Shane Greenstein. "Open Source Software and Global Entrepreneurship." *Research Policy* 52, no. 9 (2023): 104846. https://doi.org/10.1016/j.respol.2023.104846.
[35] ICANN Security and Stability Advisory Committee, "ICANN SSAC Survey on the Anticipated Impacts of Open Source Regulation on the DNS Infrastructure."

source components for critical functionality; many proprietary platforms for cloud and online services rely on one of the FOSS DNS products as a component.[36] At the same time, much of the DNS infrastructure depends on FOSS cryptographic libraries such as OpenSSL. Similarly, Linux, an open source operating system, serves as the foundation for a significant portion of cloud infrastructure, server environments, and Internet of Things (IoT) devices. This foundational role underscores the profound influence of FOSS on technological innovation. Proprietary software often incorporates these FOSS frameworks while building closed-source applications or features on top of them, which creates a symbiotic relationship between the two models.[37]

3.4.5 Contribution to Digital Autonomy

As the use of FOSS does not generally require paying licenses or compensations, the amount of capital necessary to start new digital ventures can be lower than if using proprietary software. Moreover, the freedom to study, change, and redistribute code, granted by FOSS, enables the growth of local knowledge and skills, leading to new research and business projects. Local customizations or further developments do not require any authorization or agreement by the original authors. Any service based on FOSS cannot be blocked by foreign copyright owners, as FOSS licenses cannot be revoked; even in case of trade conflicts and embargoes, the code will remain available.

3.5 Inherent Risks of the FOSS Model

While the FOSS model provides significant strengths, its unique characteristics also present a distinct set of risks that differ from those of traditional proprietary software. These are not flaws in the model itself, but rather inherent trade-offs that must be understood and managed, particularly when the software is used in critical infrastructure. The economic model, the distributed nature of development, and the lack of traditional contractual relationships all have implications for long-term sustainability and operational security. The following sections describe these inherent risks in detail.

3.5.1 Financial Sustainability and Maintainer Burnout

One of the primary threats to the DNS is the general financial sustainability problem of FOSS. The model decouples funding from use (Section 3.2.3), which enables widespread adoption, but

[36] Currently, there is no sole independent citation that supports this assertion. However, based on the investigation done collectively by the authors in this paper, we can confidently state this to be the case. There are confidentiality issues of disclosing publicly exactly which platform uses what service, and thus this is a well-founded statement without proper documentation.

[37] Gortmaker, Jeff. “Open source software policy in industry equilibrium.” Working paper, Tech. Rep, 2024. https://jeffgortmaker.com/files/Open_Source_Software_Policy_in_Industry_Equilibrium.pdf.

also allows for free riding,[38] where the whole world may depend on software that is funded by few.[39,40]

As a report from the Broadband Internet Technology Advisory Group (BITAG) noted:

> There is a disconnect between buying network equipment with expensive support contracts and failing to fund the open source software upon which, in many cases, that very equipment is dependent. Often, technical staff at network operators are willing to support development, but their corporate structure does not let them because such software is not something one can purchase in a shrink-wrapped box with a support contract. Sponsoring the development of a feature can be problematic, because the default assumption by legal departments is that software development results in the sponsor [exclusively] owning the intellectual property, which is incompatible with the free and open source software model.[41]

While the four leading open source DNS implementations are maintained by organizations that have achieved sustained financial and organizational stability, these are all small organizations. Any one of them could be destabilized if their source of funding was threatened, or if regulation imposed burdens on them beyond their ability to absorb the associated costs. Should one of the leading open source systems become undermaintained, or completely unavailable, the impact on the DNS could be significant.

This is particularly true for foundational tools that demand consistent maintenance but are disconnected from the primary business processes of their users. The four most popular open source DNS implementations are supported by development teams of no more than a dozen or so engineers, so adding project requirements that might consume a full-time staff equivalent, for example, would be a substantial burden.

Paid maintainers are funded in a variety of ways, some of which might be unexpected. Besides accepting donations and offering private technical support, some projects offer financial supporters early access to bug fixes, support for end-of-life branches, priority feature

[38] Johnson, Justin Pappas. "Essays in Microeconomic Theory." Dissertation, Massachusetts Institute of Technology, 1999. http://hdl.handle.net/1721.1/9518.

[39] Kristoff, John, Alexander Band, Ondrej Filip, and Jeff Osborn. *Panel: Core Systems OSS Panel*. NANOG 84, 2022. https://www.youtube.com/watch?v=vWiW-3jMw7w.

[40] For example, one report highlights the imbalance between the use of essential routing software and financial contributions to its development: "We illustrate the relevance of this imbalance between funding and use ... with two examples. Out of an estimated 2,000 installations of the Routinator RP software and 1,400 networks using the Krill delegated CA software, fewer than ten fund their development. The majority of operators do rely on the stability, continued existence and future development of the software, but do not contribute to this end." Broadband Internet Technical Advisory Group (BITAG) Technical Working Group, "Security of the Internet's Routing Infrastructure," 2 November 2022, 26, https://www.bitag.org/documents/BITAG_Routing_Security.pdf.

[41] BITAG Technical Working Group, "Security of the Internet's Routing Infrastructure," 26.

development, access to pre-compiled packages and additional security services.[42,43,44,45] Regulators should be careful not to inadvertently prohibit any of these as possible avenues for FOSS maintainers to fund their operations.

FOSS for DNS is exceptional in the sense that long-lived organizations employing maintainers exist at all. For their continued existence, the risk is *financial* sustainability. But in the general case, the risk is motivational. The overwhelming majority of FOSS is maintained by a single individual, in their spare time.[46] Examples for DNS software include, the DNS software most often embedded in the small, inexpensive routers used in homes and small businesses, Dnsmasq. It is mostly maintained by a single mostly unpaid volunteer.[47] Likewise, several mature software libraries for DNS programming are effectively maintained by a single individual in their spare time.[48] Another popular DNS implementation, CoreDNS and the underlying Go DNS library, are supported respectively by community efforts and a single (expert) volunteer maintainer. These examples underscore a defining characteristic of the software ecosystem: a small organization or single developer can play a critical role writing software for a world-wide digital infrastructure, while being completely decoupled from the capital or operational expenditure of operating it. For these voluntary solo maintainers, unless this work becomes funded as their primary, full-time job, the risk is not financial sustainability, but burnout. Regulators should be careful to impose additional burden beyond the ability and willingness of maintainers to invest spare time in maintenance of what are essentially hobby projects. This has been an active area of study and policymakers would do well to take note of the recommendations made by scholars.[49]

3.5.2 Supply Chain Risk from Shared Dependencies

A great deal of critical software, including both FOSS and commercial software, relies on some of the same FOSS components. A serious bug in a popular FOSS cryptography library[50] could

[42] For example, the Internet Systems Consortium offers an Early Vulnerability Notification service. See Internet Systems Consortium, "Early Vulnerability Notification (EVN)," https://www.isc.org/evn/.
[43] NLnet Labs provides, through their subsidiary Open NetLabs, training, consulting, technical support and software development services. See NLnet Labs, "Open Netlabs Services - Consultancy," https://nlnetlabs.nl/services/consultancy/.
[44] PowerDNS offers a variety of services and products including PowerDNS Protect, a security service that includes protective blocklists. See "PowerDNS Protect," https://www.powerdns.com/powerdns-protect.
[45] For an overview of the variety of kinds of open source organizational models, see "Open Source Archetypes: A Framework For Purposeful Open Source," Open Tech Strategies, Mozilla Corporation, 28 October 2019, https://blog.mozilla.org/wp-content/uploads/2018/05/MZOTS_OS_Archetypes_report_ext_scr.pdf.
[46] Bressers, Josh. "Open Source Is One Person." Open Source Security, 28 August 2025. https://opensourcesecurity.io/2025/08-oss-one-person/.
[47] "Dnsmasq - Network Services for Small Networks," https://thekelleys.org.uk/dnsmasq/doc.html.
[48] Examples include *Net::DNS* for the Perl programming language and *miekg/dns*, a Go DNS library used in Kubernetes, Docker, and the Let's Encrypt certificate authority. More examples of libraries can be found in Table 7.
[49] Eghbal, Nadia. "Roads and Bridges: The Unseen Labor Behind Our Digital Infrastructure." Ford Foundation, 14 July 2016. https://www.fordfoundation.org/work/learning/research-reports/roads-and-bridges-the-unseen-labor-behind-our-digital-infrastructure/.
[50] OpenSSL is a canonical example of a widely depended-upon cryptography library. To illustrate the scale of this dependency, at the time of the "Heartbleed' vulnerability in OpenSSL, an estimated 17% of the Internet's web servers certified by trusted authorities were believed vulnerable to attack. Netcraft, "Half a Million Widely Trusted

impact all of the DNS implementations.[51] There are other, less visible, commonly reused components as well. This is a significant potential weakness, not only in the DNS, not only in FOSS, but for software in general.[52] The sustainability risks discussed above in Section 3.5.1 apply to these shared components as well, compounding the threat.

This risk is compounded by the fact that FOSS contributors with malicious intent may be able to incorporate malicious code into open source projects or their components without detection. They can be introduced through tampering with package repositories or as contributions to projects. Vigilance in vetting and monitoring code dependencies is necessary to prevent incorporating malicious code into software projects.

3.5.3 Cost-Free FOSS Comes With No Warranty, No Guarantee of Support

Freely available software is an appealing choice since it is provided at no charge, usually with no more work than a package download. But since it comes with no warranty and no promise of support, its use also involves risks. By default, there is no contract other than the open source license agreement between the provider of the FOSS product and the user.[53] Unless the user makes their own maintenance arrangements with the maintainer or others, they have no guarantee of active maintenance or support.

If a product has security flaws, whether introduced accidentally as in the OpenSSL Heartbleed[54] bug, or maliciously as in the xz Utils backdoor,[55] nobody is required to provide fixes. While technically anyone, including the user, may take a copy of the source code and correct the problem manually, such "forking" requires significant software engineering expertise and operational resources, so that a user in most cases effectively has no control over the availability of fixes. This problem is particularly serious in packages which, like OpenSSL and xz, are so widely used that there are no easily available substitutes. The risks of monoculture are not isolated to open source, but this may be a reason why those general risks are applicable to FOSS.

A drop in community interest in providing maintenance for FOSS might appear as an incentive for the entrance of organizations that offer maintenance under contract. However, changes to the code base by these support organizations would normally be submitted back upstream to the FOSS project to minimize the burden of maintaining a separate code base. In turn, this allows

---

Websites Vulnerable to Heartbleed Bug," *Netcraft News*, 8 April 2014, https://web.archive.org/web/20141119102520/http://news.netcraft.com/archives/2014/04/08/half-a-million-widely-trusted-websites-vulnerable-to-heartbleed-bug.html.

[51] Like DNS software, developing cryptographic software is specialized work. Though some long-lived FOSS cryptography exists, diversity and quality is an issue.

[52] For an example of one attempt at making visible shared dependencies, see "Census III of Free and Open Source Software," *The Linux Foundation*, December 2024: https://www.linuxfoundation.org/research/census-iii

[53] The existence of contractual relationships between maintainers and operators is a common misconception, addressed in Section 3.2.2.

[54] "Heartbleed Bug," https://heartbleed.com/.

[55] Goodin, Dan. "What We Know about the xz Utils Backdoor That Almost Infected the World." Ars Technica, 1 April 2024. https://arstechnica.com/security/2024/04/what-we-know-about-the-xz-utils-backdoor-that-almost-infected-the-world/.

others to gain the benefit of this maintenance without paying themselves. This leads to a free rider problem: when a service provider uses FOSS but does not contribute to its maintenance, it may do so at lower cost than a competitor that does so, undercutting them.[56]

The lack of inherent warranty or support can be mitigated. Operators can train and retain on-staff expertise, fund or hire maintainers,[57] or enter into support contracts to gain access to expertise from maintainers or expert contributors.[58] The BITAG report quoted in Section 3.5.1 exemplifies organizational hurdles that keep operators from doing so.

3.5.4 Operational Risks in Deployment

The distributed nature of FOSS creates operational challenges related to software authenticity, patching, and a shortage of skilled operators.

Checking Software Authenticity

FOSS is freely distributed over the Internet, and therefore could be subject to hijacking or replacement by counterfeit code. Because of this risk, FOSS is routinely signed by the maintainer, using strong cryptographic methods. These signatures can be verified by the user to ensure the software's integrity. All of the most popular FOSS DNS implementations used in the Internet infrastructure are signed. Users, however, may not check these signatures, or may obtain the software via intermediaries who do not verify or otherwise maintain the software integrity.

Lack of Information About Deployments and Patching Speed

There is strong interest from researchers in looking for vulnerabilities in the FOSS DNS software and a strong history of them finding and reporting these. Each of the leading DNS software vendors follows generally-accepted best practices for handling, fixing and disclosing software vulnerabilities. However, there is no reliable source of information about whether, and how quickly, users are updating their software to the current, improved versions.

In FOSS in general, there is limited data to work with to assess this risk. The fact that the software consumers often receive the software via a third party—an operating system packager, or a FOSS enthusiast who may have copied and is redistributing the software—makes tracking deployment of updates even harder. This risk could be addressed by regulations requiring some sort of centralized reporting by operators on any software that is critical to their production operations.

---

[56] For example, a German business association for FOSS developers flagged this as an issue in public procurement of FOSS. See Procurement Working Group at the Open Source Business Alliance, "Selection Criteria for the sustainable Procurement of Open Source Software," Open Source Business Alliance, 11 February 2025, https://osb-alliance.de/publikationen/veroeffentlichungen/selection-criteria-for-the-sustainable-procurement-of-open source-software.

[57] Valsorda, Filippo. "I'm Now a Full-Time Professional Open Source Maintainer," 2 February 2023. https://words.filippo.io/full-time-maintainer/.

[58] Support contracts are available for several of the FOSS systems covered in this report.

Outsourcing and Skills Shortage

The complexity of correctly operating FOSS is a problem, as there is a shortage of skilled operators in some parts of the world. This combination of criticality and complexity leads some decision makers to outsource to cloud services, which may weaken the diversity and distributed nature of the system, potentially reducing resilience and security. For example, if everybody in a region chooses a popular hosted email provider because it's easier than building, operating and maintaining their own email servers, availability of email within the whole region is determined by the stability of a single mail provider.

There is also a generational shift in enterprises and smaller service providers, where earlier-career staff have become reliant on cloud services and are less familiar with operating their own network services. Several organizations run training events for operators (APNIC, NSRC, PCH) to develop DNS and other network management skills. FOSS projects are attempting to address this with pre-compiled packages and ease-of-use features, but it is still hard to compete with software-as-a-service alternatives.

4 Prevalence of FOSS in DNS and Domain Name Registration Infrastructure

FOSS plays a crucial and dominant role in the technical operations of the Internet's domain name and registration systems. In the most critical parts of this infrastructure, FOSS is the norm, and proprietary software is the exception. This report's research establishes that the global and distributed DNS infrastructure depends on FOSS. At least nine out of the 12 operators of the Internet's RSS exclusively use FOSS DNS implementations. Similarly, nine of the 10 largest service providers for TLDs use FOSS. In the domain name registration space, while many large systems are proprietary, they are overwhelmingly built upon FOSS components, and major providers of data escrow services for registries and registrars are also built upon FOSS. The following sections detail the prevalence of FOSS in each component of this critical infrastructure. The methodology is described in Appendix B.

4.1 FOSS in Domain Name Registration Infrastructure

Registration infrastructure refers to the systems which facilitate the registration of individual domain names, and which makes registered domain names available in the public DNS. While explicit data on the full extent of FOSS use is not available, the evidence shows a deep reliance on FOSS, both as complete systems and as foundational components.

A number of registry operators maintain registration infrastructure that is entirely FOSS (Table 1). For example, the FRED registry platform is known to be used by at least 12 ccTLD registries, and the Nomulus platform is used for multiple gTLD registries.

Table 1: FOSS Systems Used for Registry Operations

Software System	Open Source License	Used By
FRED	GPLv3	FRED is used by (at least) the ccTLDs for Albania, Angola, Argentina, Bosnia and Herzegovina, Costa Rica, Czech Republic, Lesotho, Macau, Malawi, North Macedonia, Paraguay, and Tanzania.
Internet ee domain registry	MIT	ccTLD for Estonia (.ee).
Namingo	MIT	Namingo is designed for the 2026 ICANN New gTLD Program: Next Round.
Nomulus	Apache 2.0	Google gTLD registries, including .app, etc.

Table 2: Registry Systems Built on FOSS Components

Registry System/ Backend Service	Example Open Source Components in Use	Registry in Use By
Afnic	Web server, Database	20 TLDs, including .fr
CIRA / SIDN / Hello Registry	Database, Web server, Application server, Reporting, Front end	6 ccTLDs, 6 gTLDs, including .ca, .ie
CoCCA	Web server, Application server, database	56 ccTLDs
CORE Association	Web Server, Database, Java Libraries	1 ccTLD, 21 gTLDs
GoDaddy Registry	Database, App servers, Reporting, Monitoring, Logging, Testing, Frontend, DNS	200+ TLDs
Identity Digital	Database	250 gTLDs and ccTLDs including ccTLDs .au, .me,.pr
Nominet	Database, Application Server, Parsing, Logging, Testing, Frontend, Nameserver	85+ TLDs, including .uk
TANGO Registry Services	Web Server, Database, Java Libraries	8 gTLDs
Tucows Registry	Database, DNS Software & accessory tooling, web &mail servers, message queues, infrastructure orchestration & virtualization, operating system	222 “TLDs", including SLDs managed as TLDs, including .my and com.my
Verisign[59]	(none provided)	.com, .net, .edu and other TLDs

[59] Verisign provided the following statement for this report: “Verisign uses its proprietary Advanced Transaction Lookup and Analysis System (ATLAS) DNS resolution platform. Verisign also uses a purpose-built registration and resolution infrastructure using a diverse and curated collection of commercial and open-source software components for redundancy.”

In contrast, the largest registries and registry backends often use proprietary systems. However, these systems are typically not built from scratch; they are often based on proprietary, customized extensions and integrations of components–such as databases and web servers–that are mostly FOSS (Table 2).

This reliance on FOSS also applies to data escrow services, which securely store copies of domain registration data. The three largest providers of these services for both registries and registrars have built their systems at least partially on FOSS components (Table 3).

Table 3: FOSS in Data Escrow Agents

Data Escrow Agent	For Registries	For Registrars	Incorporates FOSS in key functions?
Beilong Zedata (Beijing) Data Technology Co., Ltd	✓	✓	No
China Internet Network Information Center (CNNIC)	✓	✓	No
China Organizational Name Administration Center (CONAC)	✓	✓	No
DENIC Services GmbH & Co. KG	✓	✓	Yes, partially
Escrow4All	✓		Yes
Joint Stock Company Internet Exchange "MSK-IX"	✓	✓	Unknown
NCC Group	✓		Yes, partially
Taiwan Network Information Center (TWNIC)	✓		No

Registrar infrastructure was not measured or surveyed for this report. We consider it likely that registrars regularly use proprietary, customized extensions and integrations of components that are mostly FOSS. For example, some registrars use the nginx or Apache web server software to operate their web portals for registrants. They also use FOSS solutions for monitoring software, data analysis, code management, and other needs.

4.2 FOSS in DNS Publication Infrastructure (Authoritative Servers)

The evidence of FOSS dominance is most clear in the infrastructure that publishes domain information. At the highest levels of the DNS hierarchy—the root and the TLDs—FOSS is nearly ubiquitous.

The RSS is the topmost level of the DNS hierarchy. Of the 12 independent organizations that operate the root servers, at least nine exclusively use FOSS DNS implementations to answer queries (Table 4).

Table 4: FOSS Use in the Root Server System

Root Server Identifier	Software	Open Source[60]
A, J	ATLAS (proprietary) NSD	Partially[61]
B	Knot BIND9	Yes[62,63]
C	BIND9	Yes
D	NSD	Yes
E	*unknown* (FOSS), *in-house* (proprietary)[64]	Partially[65]
F	BIND9, *in-house* (proprietary)[66]	Partially
G	BIND9	Yes
H	NSD	Yes
I	*confidential*	Yes[67]
K	BIND9 Knot NSD	Yes
L	Knot NSD	Yes
M	BIND9	Yes[68]

[60] Willem Toorop et al., "RSSAC028 Implementation Study Report" (report, NLnet Labs and Stichting Internet Domeinregistratie Nederland (SIDN), September 27, 2023), 15, https://www.icann.org/en/system/files/files/rssac028-implementation-study-report-27sep23-en.pdf.

[61] From Verisign's Statement on RSSAC001v2 Service Expectations of Root Servers, "Verisign uses two diverse code bases for DNS root service: (1) our proprietary and patented ATLAS resolution platform, and (2) the open source NLnet Labs Name Server Daemon (NSD) software. One, the other, or both implementations may be in use at any given time," https://a.root-servers.org/aroot-rssac001v2-expectations.pdf.

[62] "USC/ISI's DNS Root Server," https://b.root-servers.org/.

[63] "RSSAC023v2: History of the Root Server System." ICANN Root Server System Advisory Committee (RSSAC), 17 June 2020. https://itp.cdn.icann.org/en/files/root-server-system-advisory-committee-rssac-publications/rssac-023-17jun20-en.pdf.

[64] Grant, Dani. "Delivering Dot." The Cloudflare Blog, 10 September 2017. https://blog.cloudflare.com/f-root/.

[65] Bischof, Ralph. :":E-Root Instance in San Francisco Servfails?," 19 June 2025. https://lists.dns-oarc.net/pipermail/dns-operations/2025-June/022899.html.

[66] Grant, "Delivering Dot."

[67] Shared with permission from Netnod in correspondence with SSAC dated 13 December 2024.

[68] "M-Root DNS Server," https://m.root-servers.org/.

When looking at both ccTLDs and gTLDs combined, we find that nine of the top 10 operators providing authoritative service for TLD registries use FOSS to do so.[69]

Below the root and TLDs, authoritative nameservers are operated by a vast range of entities, including individuals, enterprises, universities, and governments. While comprehensive data is not available for this entire diverse group, it is known that many of the same FOSS systems used at the root and TLDs are also the most popular choices for these operators. Many of the organizations offering secondary DNS are the same that deliver authoritative DNS services for TLDs. Table 5 and Table 6 lists FOSS systems and commercial products that are suitable for authoritative DNS implementations.

Authoritative nameservers are frequently integrated with provisioning systems to facilitate easy zone information updating and to implement proper authorization and controls over the maintenance of these records. Some of the more popular provisioning systems are FOSS.[70]

Much of the most popular content on the Internet is hosted in a few large content networks, such as Google's YouTube, which uses a proprietary DNS system. While a number of large operators of authoritative services for second and lower levels in the hierarchy use FOSS (for example, those that also serve the root or TLD zones), not enough public statements are available to reliably survey and produce statistics with regard to their use of FOSS.[71] See tables below for more information. This is a notable gap in the information available to us, because four very large providers may account for over half of the queries for authoritative names visible on the Internet.[72]

4.3 FOSS in DNS Retrieval Infrastructure (Resolvers)

The prevalence of FOSS is not limited to the publication of DNS data; it is equally significant in the infrastructure that retrieves that information: the diverse ecosystem of DNS resolvers. FOSS has a substantial presence across the entire resolver ecosystem, from local networks to global cloud platforms.

Most users are served by local resolvers operated by their ISPs, enterprises, or educational institutions. Research has estimated the total percentage of users served by the cloud resolvers to be less than 20% worldwide. The remaining 80% of users are using some sort of local resolver.[73]

---

[69] Top operators by number of TLDs served. See Appendix B for a description of the methodology used.

[70] Popular systems include VinylDNS (https://www.vinyldns.io/) which is maintained by Comcast, and OctoDNS, which is maintained by Amazon and Oracle. DNS Control is another popular configuration system, that manages DNS across both self-hosted systems and cloud services including Cloudflare, Amazon's Route53 service, and Gandi, a DNS registrar and hosting provider.

[71] Reliable adoption measurements of FOSS is a hard problem. Daniel Stenberg offers several reasons why this is the case in "What We Can't Measure," Daniel://Stenberg:// (blog), 5 June 2025, https://daniel.haxx.se/blog/2025/06/05/what-we-cant-measure/.

[72] Huston, Geoff. "Looking at Centrality in the DNS." APNIC Blog (blog), 22 November 2022. https://blog.apnic.net/2022/11/22/looking-at-centrality-in-the-dns/.

[73] Huston, 'Looking at Centrality in the DNS'.

Many of the most common FOSS systems can be used for both authoritative and resolver functions (Table 5).

Table 5: Commonly Used FOSS Systems for DNS Server Applications

Software System	Open Source License	Application - Example Users
BIND9	MPL 2.0	Authoritative, Resolvers - CIRA, NIC.BR, Visionary Broadband
CoreDNS	Apache 2.0	Kubernetes, Authoritative - Meta
dnsdist	GPL 2.0	DNS load-balancing
Dnsmasq	GPL 2 or 3	Predominantly Resolvers - popular in embedded systems, such as OpenWRT, home gateways.
Knot DNS	GPL 3.0	Authoritative - .cz TLD
Knot Resolver	GPL 3.0	Resolver - DNS4EU
NSD	BSD 3-Clause	Authoritative - Rcode Zero
PowerDNS	GPL 2.0	Authoritative - Rakuten
PowerDNS Recursor	GPL 2.0	Resolver - British Telecom
Unbound	BSD 3-Clause	Resolver - Quad9, Let's Encrypt
YADIFA	BSD 3-Clause	Authoritative - .eu TLD

While many commercial products exist for this market, most of them incorporate one or more FOSS solutions as the core DNS component of their offering (Table 6).

In the cloud, hyperscale computing platforms such as Microsoft Azure, Google Cloud, and Amazon's AWS all operate significant resolver infrastructure to support their services. At least four of the biggest hyperscalers rely on FOSS for DNS resolving,[74] while others have built proprietary solutions based on FOSS DNS libraries (Table 7).

Finally, some end users configure their systems to bypass the resolver provided by their network operator and instead use open public resolvers. While two of the most popular public services (Google's 8.8.8.8 and Cloudflare's 1.1.1.1) use proprietary software, other major public resolvers, such as Quad9 (9.9.9.9) and DNS4EU, are built on FOSS. More detail is available in Appendix B.

[74] Vague by design based on confidentiality.

Table 6: Examples of Commercial DNS Services Incorporating FOSS

Manufacturer	Product	Application	Incorporates FOSS
Akamai	Edge DNS	Hybrid cloud resolver and authoritative service	No
Bluecat Networks	Integrity, Micetro	Authoritative, resolver	Yes
Cygna Labs	VitalQIP, DiamondIP	Authoritative, resolver	Yes
EfficientIP	SolidServer DDI	Authoritative, resolver, appliance and cloud	Yes
F5	BIG-IP DNS	Resolver	Yes
IBM	NS1 Connect	Cloud-based authoritative service	Unknown
InfoBlox	Universal DDI & NIOS DDI	Authoritative, resolver, appliance and cloud	Yes
Knipp	IronDNS	Authoritative	Partial
Microsoft	Windows Server DNS	Authoritative, resolver, used in enterprise networks, integrates with Active Directory	No
	Azure DNS	Cloud resolver service	Yes
Netgate	pfSense	Resolver	Yes
Oracle	OCI DNS	Authoritative cloud service	Yes
TCPWave	DDI Management	Authoritative appliance	Yes

Table 7: FOSS Libraries Used for DNS Infrastructure Applications

Software System	Programming Language	Application - Example Users
c-ares	C	libcurl, curl, NodeJS
dnsjava	Java	(Proprietary) registry backends
dnspython	Python	Mailman, Samba, Ansible
domain	Rust	Cascade
miekg/dns	Go	Let's Encrypt, CoreDNS, Docker
ldns	C	Zonemaster, dnstap, (proprietary) nameservers
libunbound	C	Open vSwitch, libreswan, opendkim
Net::DNS	Perl	Spamassassin, Mail::DMARC, Mail::DKIM, Mail:SPF

5 Contemporary cases in FOSS Regulation

This section examines several contemporary cases from the United States, United Kingdom, and European Union that illustrate how policymakers are adapting cybersecurity regulations to the unique realities of the FOSS ecosystem. Table 8 provides a high-level summary of these approaches, which show a pattern of exempting volunteer maintainers from direct liability while focusing on the responsibilities of commercial entities who integrate or deploy FOSS. Each case is then explored in detail.

Table 8: Summary of Contemporary Regulatory Approaches to FOSS

Section	Key Focus	Example Regulation	FOSS Treatment
5.1	Allocate responsibility	2023 US Cybersecurity Strategy, 2025 UK Code	Exempt maintainers from liability, focus on commercial entities
5.2	Incentivize collaboration	EU Cyber Resilience Act	Introduce optional 'steward' role to incentivize support
5.3	Avoid proprietary assumptions	EU NIS 2 Implementing Act	No contracts = No direct supplier, encourage support
5.4	Avoid conflicting regimes	EU NIS 2 Directive	Avoid overlaps for global elements like root servers

5.1 Allocate responsibility to stakeholders most capable of action

The 2023 US National Cybersecurity Strategy[75] sought to shift liability for insecure software products and services (strategic objective 3.3). It formulated a high-level vision wherein "companies that make software must have the freedom to innovate, but they must also be held liable when they fail to live up to the duty of care they owe consumers, businesses, or critical infrastructure providers." From the outset, the strategy singled out FOSS maintainers for a more nuanced treatment: "Responsibility must be placed on the stakeholders most capable of taking action to prevent bad outcomes, not on the end-users that often bear the consequences of insecure software nor on the open source developer of a component that is integrated into a commercial product."

Similarly, the UK's 2025 voluntary Software Security Code of Practice[76] aims to "support software vendors and their customers in reducing the likelihood and impact of software supply chain attacks and other software resilience incidents." It is similarly scoped towards commercial software vendors: "For open source software, the developer/maintainer bears no formal commitment to their onwards supply chain or for the ongoing maintenance and security of their

[75] US President. "National Cybersecurity Strategy." Washington, DC: The White House, 1 March 2023. https://bidenwhitehouse.archives.gov/wp-content/uploads/2023/03/National-Cybersecurity-Strategy-2023.pdf.
[76] "Software Security Code of Practice."

code. Any risks associated with open source code must be managed by end-users or proprietary developers using open source code in their software."

5.2 Incentivize cross-industry collaboration on sustainable maintenance

The EU CRA[77] aims to address a "low level of cybersecurity of products with digital elements, reflected by widespread vulnerabilities and the insufficient and inconsistent provision of security updates to address them." While similarly carving out from responsibility FOSS maintainers that do not monetize FOSS, it incentivizes cross-industry collaboration on sustainable maintenance of FOSS by introducing a new legal actor ("open source software steward") that "provides support on a sustained basis for the development" and ensures "the viability of those products." Stewards are an optional, not yet widespread, role for organizations that FOSS maintainers can choose to associate with as a means to funnel resources from manufacturers or operators of essential infrastructure dependent on FOSS. While the amount of FOSS currently supported by a steward-like organization is currently tiny, the CRA may increase this practice going forward, and it is likely applicable for some of the organizations maintaining DNS software. A final regulatory innovation is the future option of "voluntary security attestations," allowing cross-industry collaboration on due diligence efforts.

5.3 Avoid supply chain security requirements assuming proprietary software

The EU NIS 2 Directive seeks to "mitigate threats to network and information systems used to provide essential services in key sectors."[78] It regulates Digital Infrastructure as a "Sector of High Criticality," assigns management responsibility for cybersecurity and mandates "risk-management measures," detailed in an implementing act.[79] These include "supply chain security, including security-related aspects concerning the relationships between each entity and its direct suppliers or service providers." The Annex that spells out these requirements is derived from controls in ISO/IEC 27002:2022, and contrary to the reality of FOSS (see Section 3.2.2) assumes a chain of contractual obligations up to the software developer.

[77] Regulation (EU) 2024/2847 of the European Parliament and of the Council of 23 October 2024 on horizontal cybersecurity requirements for products with digital elements and amending Regulations (EU) No 168/2013 and (EU) 2019/1020 and Directive (EU) 2020/1828 (Cyber Resilience Act), 2024 O.J. (L 2024/2847) 1, https://eur-lex.europa.eu/eli/reg/2024/2847/oj/eng.

[78] Directive (EU) 2022/2555 of the European Parliament and of the Council of 14 December 2022 on measures for a high common level of cybersecurity across the Union (NIS 2 Directive), 2022 O.J. (L 333) 80, https://eur-lex.europa.eu/eli/dir/2022/2555/oj/eng.

[79] Commission Implementing Regulation (EU) 2024/2690 of 22 October 2024 laying down rules for the application of Regulation (EU) 2024/2847 of the European Parliament and of the Council, 2024 O.J. (L 2024/2690) 1, https://eur-lex.europa.eu/eli/reg_impl/2024/2690/oj.

In its technical implementation guidance for regulated entities,[80] the European Union Agency for Cybersecurity, ENISA, clarifies the notion of "direct supplier and service provider" with respect to FOSS: "In the case of free and open source software (FOSS), communities and projects that openly develop, maintain and distribute software may not be considered direct suppliers or service providers where no contractual relationship exists between the relevant entity and the open source project, beyond adherence to a standardised copyright licence, or where the contractual relationship is with an open source software steward." Instead, it recommends to "Consider supporting the communities developing and maintaining FOSS and invest in a mutually beneficial relationship with them. Where effective, this could involve relationships with the relevant OSS steward that 'provid[es] support on a sustained basis for the development and ensures the viability of those products.'"

A 2025 report commissioned by the UK's Department for Science, Innovation and Technology surveyed industry practices for open source and for supply chain risk.[81] Amongst other things, the report recommends that organizations "Establish an internal OSS policy to manage the adoption of OSS components." and "Promote active engagement with the OSS community to [..] ensure high-quality OSS components and a sustainable OSS ecosystem." In contrast, a supply chain security regulation not tailored to FOSS would impose contractual requirements on operators to require background checks of FOSS maintainers with the view that they are "suppliers," even though FOSS is not purchased. Such ill-considered regulation might cause FOSS talent to abandon their projects, leaving projects with fewer or less competent maintainers, causing quality to deteriorate. This runs counter to the intended effect of supply chain security regulation.

5.4 Avoid conflicting regional regimes for global FOSS communities

Direct regulation of software development and its use in critical infrastructure is not a widespread practice. Because FOSS allows for global collaborative development (Section 3.2.1), future regulatory regimes should avoid creating overlapping and mutually conflicting requirements differentiated by the physical location of individual maintainers.

The aforementioned EU NIS 2 Directive[82] took great care to avoid a similar situation of overlapping regimes with its requirements for DNS service providers by excluding the root name servers from its regulatory scope, thereby avoiding a situation whereby the Root Server Operators would be subject to overlapping, sometimes conflicting regional regimes.[83]

[80] European Union Agency for Cybersecurity (ENISA), NIS2 Technical Implementation Guidance (report, Publications Office of the European Union, 2025), https://www.enisa.europa.eu/publications/nis2-technical-implementation-guidance.
[81] "Open Source Software Best Practices and Supply Chain Risk Management."
[82] NIS 2 Directive.
[83] A summary of arguments by Bert Hubert can be found in "Dear EU: Please Don't Ruin the Root," 10 May 2021, https://berthub.eu/articles/posts/dont-ruin-the-root/.

6 Key Findings

This section consolidates the report's core analysis into a series of findings that form the evidentiary basis for the actionable guidelines that follow.

Finding 1: FOSS is the foundation of critical DNS infrastructure. As policymakers and regulatory bodies around the world seek to secure the software supply chain, it is critical that their efforts are informed by a clear understanding of how the Internet's most foundational systems are actually built and maintained. Research for this report establishes that the global DNS infrastructure today is overwhelmingly dependent on FOSS. In the most critical parts of this infrastructure, FOSS is the norm, and proprietary software is the exception. This includes the RSS, where at least nine of the 12 operators use FOSS DNS implementations exclusively, and the TLDs, where nine of the 10 largest service providers use FOSS.

Finding 2: The FOSS development model is fundamentally different from proprietary software. While proprietary software is typically created internally by a single organization, the FOSS model is open and distributed. It is built upon four essential freedoms granted by its licenses: the freedom to use, study, share, and change the software. This framework fosters a unique ecosystem of maintainers, contributors, and operators who are generally not restricted by the contractual relationships that define a traditional commercial supply chain.

Finding 3: FOSS is not inherently more or less secure than proprietary software; security depends on the process and maintenance, not on the visibility of the source code. The security implications of this open model must be understood in the context of a decades-old debate that has no clear answer. As security engineering researcher Ross Anderson summarized, for large and complex systems, the visibility of source code helps both attackers and defenders equally. In the long run, whether a system is open or closed makes little difference to its security. Hiding source code does not provide additional security; rather, a system's security emerges from the quality of its development, review, and maintenance processes.

Finding 4: The DNS FOSS ecosystem has unique strengths that promote stability and resilience. While openness itself may be neutral, the collaborative process it enables is particularly effective for building and maintaining critical Internet infrastructure. The inherent transparency of FOSS allows a global community of developers, researchers, and operators to study the source code and address vulnerabilities collectively, a practice that often results in quicker patching than in proprietary systems. This leads to inherent strengths, including enhanced collaborative security, operational resilience through software diversity, and remarkable stability provided by long-term support from dedicated non-profit and commercial organizations.

Finding 5: The FOSS model has inherent risks that require a tailored, not a one-size-fits-all, policy approach. The same characteristics that provide these strengths also introduce a distinct set of risks that require a tailored policy approach. Because the FOSS development model decouples funding from use, projects can face challenges with financial sustainability and

maintainer burnout, where critical infrastructure may depend on the unfunded, voluntary efforts of a few individuals. Furthermore, the widespread reuse of FOSS components creates a risk of shared dependencies, where a vulnerability in a single library can have a cascading, ecosystem-wide impact. These issues cannot be solved by regulations designed for a commercial, proprietary software market.

Finding 6: Attaching new legal and financial liability to the development and distribution of FOSS must be done carefully to avoid poisoning the environment which has led to the development of this fundamental element of Internet infrastructure. The lack of a single, accountable legal entity or a conventional contractual chain in FOSS projects makes it both impractical and risky to apply traditional liability frameworks. The FOSS development model depends on individual volunteers and small, minimally funded organizations. Imposing heavy regulatory burdens on these maintainers could discourage their involvement, potentially stifling innovation and leading to the abandonment of software that is crucial for critical Internet infrastructure.

Finding 7: FOSS is important to enabling new providers to enter the Internet services market and presents opportunities for Policymakers to encourage development of local services and reduce dependence on foreign cloud providers. Beyond its technical role, FOSS is an enabler of economic growth, digital autonomy, and market competition. The FOSS development model lowers the cost of market entry for new businesses by eliminating software licensing fees. This accessibility nurtures local innovation and skills, giving policymakers a powerful way to create a more diverse digital ecosystem and can help reduce dependence on foreign cloud services.

7 Actionable Guidelines for Policymakers

This section builds upon the report's findings to provide direct, actionable guidelines for policymakers. The goal is to enable the development of effective, non-harmful regulation that strengthens, rather than undermines, the FOSS ecosystem that is critical to the secure and stable operation of the Internet.

Guideline 1: Acknowledge the Critical Role of FOSS. This report establishes the fact that the global DNS infrastructure is dependent on FOSS. In the most critical parts of this infrastructure, FOSS is the norm, and proprietary software is the exception. Therefore, policymakers should explicitly acknowledge in any relevant legislation or regulation that FOSS underpins critical Internet infrastructure and that its use is a strength to be preserved. This recognition should inform regulation starting from the initial design phase to prevent unintended harm to the ecosystem.

Guideline 2: Consult the FOSS Community. The FOSS development model is open, distributed, and fundamentally different from that of proprietary software. It is an ecosystem of maintainers, contributors, and operators who are generally not bound by the contractual relationships that define a traditional commercial supply chain. It is essential to involve all

sectors of the FOSS ecosystem—including companies, non-profits, individual maintainers, and community institutions—throughout the policy development process. This ensures that regulations are informed by their operational realities and prevents unintended harm to the FOSS ecosystem and consequently, critical Internet infrastructure.

Guideline 3: Make Use of the Contemporary Cases in FOSS Regulation. As detailed in Section 5, recent regulatory efforts have already begun to navigate the unique characteristics of FOSS while still advancing essential policy objectives. These cases provide a valuable framework for developing new policies that are tailored to the unique characteristics of the FOSS ecosystem. Applying these lessons means designing policies and regulations that:

- Allocate responsibility to stakeholders most capable of action. The duty of care is placed on the entities that deploy software in commercial products or critical infrastructure, not on the volunteer FOSS developers who create the components.
- Incentivize cross-industry collaboration on sustainable maintenance. The viability of critical FOSS products can be ensured through the support of innovative legal models, like the 'open source software steward,' which funnel resources from industry.
- Avoid supply chain security requirements that assume a proprietary software model. Recall that in FOSS, there is often no direct, contractual relationship between a maintainer and an operator beyond the open source license itself.
- Avoid creating conflicting regional regimes for global FOSS communities. Overlapping and contradictory obligations for global FOSS projects should be avoided in regulations to prevent the fragmentation of development and undermining of security.

Guideline 4: Incentivize FOSS Sustainability. The FOSS model decouples funding from use, which has been known to lead to financial sustainability challenges and maintainer burnout. Critical infrastructure may depend on small organizations or the unfunded, voluntary efforts of a few individuals. To mitigate this risk, policymakers are encouraged to create favorable policies to encourage public and private sector contributions to critical FOSS projects as a form of investment in a shared public good.

Guideline 5: Address Systemic Risks Collectively. The widespread reuse of FOSS components creates a risk of shared dependencies, where a vulnerability in a single library can have a cascading, ecosystem-wide impact in both FOSS and proprietary software products. As this is a systemic risk inherent to all modern software development, policy should foster and fund collaborative, ecosystem-wide solutions like improved security tooling and independent research, rather than placing the full burden on individual volunteer maintainers.

8 Acknowledgments, Disclosures of Interests, and Withdrawals

In the interest of transparency, these sections provide the reader with information about aspects of the SSAC process. The Acknowledgements section lists the SSAC members, outside experts,

and ICANN staff who co-authored or contributed directly to this particular document or who provided reviews. The Disclosures of Interest section points to the biographies of all SSAC members, which disclose any interests that might represent a conflict—real, apparent, or potential—with a member's participation in the preparation of this report. The Withdrawals section identifies individuals who have recused themselves from the discussion of the topic with which this report is concerned. Except for members listed in the Withdrawals section, this document has the consensus approval of all of the members of SSAC.

8.1 Acknowledgments

The committee wishes to thank the following SSAC members, invited guests, and ICANN staff for their time, contributions, and review in producing this report.

SSAC Members
Joe Abley
Maarten Aertsen (Work party co-chair)
Gautam Akiwate
Tim April
Nabil Benamar
KC Claffy
Hadia Elminiawi
Ondrej Filip (SSAC member through 31 December 2024)
James Galvin
Robert Guerra
Russ Housley
Matthias Hudobnik
Geoff Huston
Layal Jebran
Merike Kaeo (SSAC member through 31 December 2024)
Andrei Kolesnikov
Warren “Ace” Kumari
Barry Leiba (Work party co-chair)
John Levine
Russ Mundy
Ram Mohan
Matt Thomas
Peter Thomassen
Tara Whalen
Suzanne Woolf
Jiankang Yao

Invited Guests
Vittorio Bertola
Merike Kaeo (Invited Guest from 01 January 2025)
Vicky Risk
Raffaele Sommese

ICANN Staff
John Emery (Editor)
Daniel Gluck
Gustavo Lozano Ibarra
Michael Puckett
Carlos Reyes
Danielle Rutherford (Editor, Contributing Author)
Kathy Schnitt
Steve Sheng (SSAC Support Staff through 30 November 2024)

8.2 Disclosures of Interest

SSAC member biographical information and Disclosures of Interest at the time of publication are available at: https://www.icann.org/en/ssac/members/archive/16-05-2025.

8.3 Withdrawals

There were no withdrawals.

Appendix A: Glossary and Acronyms

A.1 Glossary of Terms

Authoritative Server: A server that holds the definitive, official DNS records for a particular domain name. It provides the final answers to DNS queries for that domain.

Contributor: An individual or organization that offers improvements to a FOSS project, such as by submitting code, documentation, or bug reports.

Data Escrow: Storing a copy of domain registration data at an ICANN accredited third-party for safekeeping.

Domain Name: A unique, human-readable name (e.g., icann.org) that identifies a specific address on the Internet and forms the basis of URLs.

Domain Name System (DNS): The global, decentralized system that acts as the "address book for the Internet," translating human-readable domain names into the numerical IP addresses necessary for locating computer services and devices.

Extensible Provisioning Protocol (EPP): A standardized technical protocol used to automate transactions between domain name registrars and registries, such as registrations, renewals, and transfers.

Fork: A new, separate software project that is started by taking a copy of the source code from an existing FOSS project.

Free and Open Source Software (FOSS): Software that is licensed in a way that grants users four essential freedoms: to use, study, share, and change the software. This defines a collaborative development model, not just software that is free of charge.

Internet Protocol (IP) Address: A unique numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication.

Internationalized Domain Name (IDN): A domain name in which one or more of its labels contain characters other than ASCII letters, digits, or hyphens.

Maintainer: The individual or group responsible for the overall direction and quality control of a FOSS project. They have the authority to accept or reject contributions to the official version of the software.

Operator: An individual or organization that deploys and uses software to run a service. In the context of the DNS, an operator is an entity that runs DNS infrastructure components, such as authoritative servers or resolvers.

Publication (in the DNS): The technical process of making a domain's DNS records available on authoritative servers so that the domain name can be found by others on the Internet.

Recursive Resolver (or Resolver): A server, often operated by an Internet Service Provider (ISP), that acts on behalf of a user's device to find the correct IP address for a requested domain name.

Registration (in the DNS): The administrative process of reserving a unique domain name by adding it to the authoritative master database (the registry) for a specific top-level domain.

Registrant: The individual or organization that registers and holds the rights to a specific domain name.

Registrar: A public-facing organization that acts as a retailer for domain names, managing the reservation of domains on behalf of registrants.

Registry: The authoritative, master database of all domain names registered within a specific top-level domain (e.g., the .org registry). The organization that maintains this database is the registry operator.

Root Server System (RSS): The collection of servers at the highest level of the DNS hierarchy that are responsible for directing queries to the correct top-level domain servers.

Top-Level Domain (TLD): The segment of a domain name located to the right of the final dot, such as .com, .org, or .uk.

Uniform Resource Locator (URL): The full address used to find a specific resource on the Internet, which typically includes a protocol (e.g., https), a domain name, and a specific path (e.g., https://www.icann.org/resources).

A.2 Abbreviations Used in This Report

ccTLD: Country-Code Top-Level Domain
DNS: Domain Name System
EPP: Extensible Provisioning Protocol
FOSS: Free and Open Source Software
gTLD: Generic Top-Level Domain
IP: Internet Protocol
ISP: Internet Service Provider
RSS: Root Server System
SSAC: Security and Stability Advisory Committee
TLD: Top-Level Domain
URL: Uniform Resource Locator
IDN: Internationalized Domain Name

Appendix B: FOSS Prevalence Research Methodology and Findings

This appendix provides a complete, self-contained summary of the report's original research. It details both the methodology used to obtain the findings on free and open source software (FOSS) prevalence and the findings themselves, which are presented in Section 4 of this report.

B.1 General Approach and Challenges

Determining with certainty which software operators use in the wild is challenging. While certain domain name system (DNS) records (e.g., version.bind, authors.bind, id.server) were introduced to help end-users identify the DNS server version they are communicating with, these records are not widely adopted due to potential security risks and the need for manual configuration. Other methods have been explored in the literature, including passive analysis and active measurements. However, these approaches are often limited in scope—some can only identify recursive infrastructure—and face scalability challenges.

For these reasons, the approach taken in this report focuses on evaluating large DNS operators by market share, where significant usage of open source software could be directly confirmed.

B.2 Domain Name Registration Infrastructure

B.2.1 Methodology

To assess the use of FOSS in registration infrastructure, the Security and Stability Advisory Committee (SSAC) surveyed large registries and registry backend providers. To assess the use of FOSS in data escrow services, the ICANN-approved Data Escrow Agents (DEAs) listed on the ICANN website were surveyed, with a focus on software used for key service components (data transfer, signature verification, deposit verification and interaction with the Registration Reporting Interfaces (RRI) API.

B.2.2 Findings

Registration infrastructure refers to the systems which facilitate the registration of individual domain names and make them available in the public DNS. While explicit data on the full extent of FOSS use is not available, the evidence shows a deep reliance on FOSS, both as complete systems and as foundational components.

A number of registry operators, particularly in the country-code top-level domain (ccTLD) space, maintain registration infrastructure that is entirely FOSS (Table 1, Table 2). The FRED registry platform, for example, is known to be used by at least 12 ccTLD registries, and the Nomulus platform is used for multiple generic TLDs (gTLDs).

In contrast, the largest registries and registry backend service providers often use proprietary systems. However, these systems are typically not built from scratch; they are proprietary, customized extensions and integrations of components, such as databases and web servers, that are mostly FOSS.

This reliance on FOSS extends to data escrow services (Table 3), which securely store copies of domain registration data. The largest providers of these services for both registries and registrars have built their systems at least partially on FOSS components.

B.3 DNS Infrastructure

B.3.1 Methodology

Authoritative Server Analysis: To investigate the use of FOSS in TLD infrastructure, the following procedure was used:

1. The root zone was fetched from IANA and the TLDs were mapped to the Internet Protocol (IP) addresses of their nameservers.
2. A separate analysis was performed on ccTLDs by considering both two-letter TLDs and their A-label (Punycode) equivalents (to account for Internationalized Domain Name (IDN) ccTLDs).
3. The Python ip2asn library was used to map these IP addresses to their respective Autonomous System Numbers (ASNs) and ASN names (operators).
4. Market share was computed as the number of TLDs hosted by a given operator divided by the total number of TLDs (noting that a TLD can be hosted by multiple operators simultaneously).
5. The top 25 operators were identified and their use of open source authoritative nameserver software was manually verified.

Resolver Analysis: There are no reliable surveys of the entire installed base of resolvers. For this report, analysis focused on enumerating the major operators and types of deployments (e.g., local, cloud, public) and surveying whether they used FOSS or proprietary software based on public statements and direct knowledge.

B.3.2 Findings

Authoritative Servers: The evidence of FOSS dominance is most clear in the infrastructure that publishes domain information. At the highest levels of the DNS hierarchy—the root and the TLDs—FOSS is nearly ubiquitous. The root server system (RSS) sits at the apex of the DNS. Of the 12 independent organizations that operate the root servers, at least nine exclusively use FOSS DNS implementations to answer queries (Table 4).

This pattern continues at the next level of the hierarchy with TLD nameservers (Table 5). Research for this report found that nine of the top 10 operators providing authoritative DNS service for TLD registries use FOSS to do so. Furthermore, 20 out of the top 25 operators providing this service for ccTLDs use FOSS DNS software, collectively serving 234 unique ccTLDs.

Resolvers: The prevalence of FOSS is not limited to the publication of DNS data; it is equally significant in the infrastructure that retrieves that information: the diverse ecosystem of DNS resolvers.

Most users are served by local resolvers operated by their Internet Service Providers (ISPs), enterprises, or educational institutions. While many commercial products exist for this market, most of them incorporate one or more FOSS solutions as the core DNS component of their offering.

In the cloud, hyperscale computing platforms such as Microsoft Azure, Google Cloud, and Amazon Web Services all operate significant resolver infrastructure to support their services. At least four of the biggest hyperscalers rely on FOSS for DNS resolving, while others have built proprietary solutions based on FOSS DNS libraries.

Appendix C: Survey of DNS Operator Perspectives on FOSS and Software Regulation

The Security and Stability Advisory Committee (SSAC) conducted an informal, online survey on the anticipated impacts of open source regulation on the Domain Name System (DNS) infrastructure to gather input for this report. The EUSurvey tool was used to protect the privacy and anonymity of respondents.[84] Responses were solicited from the DNS technical community on a number of mailing lists, including; the Council of European National Top-Level Domain Registries (CENTR) technical and legal mailing lists, the Réseaux IP Européens Network Coordination Centre (RIPE NCC) DNS and Open Source working groups, and user mailing lists for several open source DNS software systems. The survey was also presented at the DNS Operations, Analysis and Research Center (DNS-OARC) February meeting.

The main purpose of the survey was to discover whether technical operators in the DNS are aware of regulatory initiatives, and to ask what effects they anticipate may follow from regulation of free and open source software (FOSS). The survey was open for the month of February 2025. There were 98 responses covering a cross section of roles in the DNS infrastructure.

We first asked respondents about their involvement in the DNS. 96 of the 98 respondents reported involvement in the DNS infrastructure, and 64 respondents also reported involvement in the domain name registration infrastructure. Twenty-four respondents included "software developer / vendor" as one of their roles, but none of them were solely responding as software developers. (We did not ask whether their software was open or closed source.)

Respondents were open source users, and quite aware of regulation. Nearly all respondents reported using open source (93%). Approximately one third (33%) reported also using proprietary software. Thirty percent of respondents also advise others on the implementation and operation of FOSS. This group had a high level of awareness of current regulatory initiatives. We provided a list of regulatory initiatives and asked "Which of these regulatory initiatives are you aware of? (check all that you are aware of)." Seventy-seven respondents (so approximately 77%) indicated they were aware of one or more. The EU Cyber Resilience Act, Cyber Security Act and NIS2 were each familiar to more than 40% of respondents. Thirty percent of respondents also indicated they were familiar with one or more of the US Government initiatives, including the Executive Orders 14028 and 14144, the Secure Software Development Attestation program, or the CyberTrust Mark for Internet of Things (IoT) devices. Other regulations mentioned by respondents included the Australian Security of Critical Infrastructure Act (SOCI), General Data Protection Regulation (GDPR), Payment Card Industry Data Security Standard (PCI DSS) and US and Canadian Federal Risk and Authorization Management Program (FedRAMP) authorizations.

[84] "EUSurvey – About." https://ec.europa.eu/eusurvey/home/about.

We asked "What specific concerns do you have about the impact of software regulation on your organization?" with both positive and negative options provided. All respondents were able to answer this question, including those who did not indicate familiarity with any of the specific regulations we mentioned.

- 72% of respondents felt that it was reasonably likely that some open source projects may be abandoned, or become unavailable
- 66% thought that compliance will increase the cost of software to them
- 49% were concerned that some open source projects they rely on may switch to proprietary licensing
- 29% thought that regulation will inhibit their organization from publishing software as open source
- 21% anticipated that the security of open source they use will improve
- 7% anticipated that regulation will lower the burden of assessing software security and quality on their organizations.

Although we specifically solicited comments about positive impacts and opportunities, in an attempt at a balanced position, the respondents were mostly pessimistic.

C.1 Open-ended comments (specific concerns)

The survey invited open-ended comments on specific concerns about the impact of software regulation, as well as asking about opportunities or positive impacts anticipated.

The most frequently cited concerns were:

- Increased compliance costs
- Slower software deployment
- Potential abandonment of some open source projects due to regulatory burdens
- Increased complexity in legal compliance for open source software users