Distributing Trust in Critical Societal Scale Computing Infrastructure

Sudheesh Singanamalla

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington

2024

Reading Committee:

Kurtis L. Heimerl, Chair

Richard J. Anderson, Chair

Arvind Krishnamurthy

Tadayoshi Kohno

Program Authorized to Offer Degree:

Computer Science & Engineering

University of Washington

Abstract

Distributing Trust in Critical Societal Scale Computing Infrastructure

Sudheesh Singanamalla

Co-Chairs of the Supervisory Committee:
Kurtis L. Heimerl
Paul G. Allen School of Computer Science and Engineering

Richard J. Anderson
Paul G. Allen School of Computer Science and Engineering

Over the last decade, the world has seen an increase in the adoption of computing technology powered by the affordability of innovative mobile devices, exponential growth in access to the Internet, and cloud computing that allows organizations to easily scale their solutions worldwide. Computing is woven into the fabric of society and has begun to reshape it in unexpected ways. These societal changes have increased our reliance on hidden computing and networking infrastructure powering cloud, telecom, and Internet service providers (ISPs), making them *critical societal scale computing infrastructure*.

Today, we increasingly trust a small number of infrastructure providers, who operate at nation-scale with an incredible amount of our data and private information. While this ongoing colocation resulted in economies of scale, it opened up tremendous abuse potential. Organizations providing critical services to citizens in a country could (1) maliciously misuse the data collected without consent, (2) be legally compelled to breach user privacy by governments, (3) be attacked by hackers in efforts to breach and sell user data, and (4) mis-configure services or face infrastructure failures which might appear as attacks – affecting customer trust, cause reputation and economic damages.

While decentralization might be a tempting solution to address these challenges, it is difficult to achieve the scale, performance and ease of access of todays networks. My work presented in this dissertation focuses on scalable, and practical mechanisms in which users interacting with hidden infrastructure could gain privacy benefits keeping security unaffected through improved transparency, while maintaining comparable performance. This dissertation challenges the default trust settings in today's computing infrastructure and proposes secure practical alternatives specifically to (1) democratize Internet access, (2) enable private, and verifiable communications with critical Internet services, and (3) reduce user's trust requirements through improved transparency.

Table of Contents

List of Figures

List of Tables

Table Number Page

Glossary

4G/LTE: 4th Generation wireless long term evolution standards introduces an all-IP based network for both data and voice over the network, switching from the older circuit switched networking in 3G.

5G: 5th Generation wireless network standards which provide greater radio layer and core network flexibility.

5GC: The core network architecture for a 5G network similar to the EPC but provides service based interfaces while separating user and control planes. The 5G core network contains the AMF, UDR, UDM, AUSF and many other services responsible for enabling 5G connectivity and services.

AUTHENTICATED ENCRYPTION WITH ASSOCIATED DATA (AEAD): A set of symmetric key schemes that simultaneously assure data confidentiality and its authenticity protecting the confidentiality, integrity and authenticity.

ACCESS AND MOBILITY FUNCTION (AMF): A 5GC components responsible for managing user access and mobility in 5G networks similar to the MME in LTE networks.

AUTONOMOUS SYSTEM (AS): A large network or group of networks with a single routing policy – typically under the control of a single organization and assigned a unique number.

AUTHENTICATION SERVER FUNCTION (AUSF): A stateless 5G network function that computes and generates the required authentication vectors responsible for authenticating user devices.

BORDER GATEWAY PROTOCOL (BGP): The routing protocol for the Internet enabling various autonomous systems to exchange routing and reachability information in addition to determining the best network routes for Internet data transmission.

CERTIFICATE AUTHORITY (CA): An organization that validates and certifies the cryptographic public key identity of the requesting entity.

COMMUNITY CELLULAR NETWORK (CCN): A standalone and complete network enabling local communities to own and operate their own cellular networks – typically deployed in remote communities with no national carrier coverage.

COUNTRY-CODE TOP LEVEL DOMAIN (CCTLD): A TLD that indicates the country of the domain – typically reserved and managed by each country or territory eg. com.uk, gov.in etc..,

CONTENT DELIVERY NETWORK (CDN): A globally distributed network of caches and server infrastructure physically located closer to the end users reducing latency to information access through content replication.

CERTIFICATE TRANSPARENCY (CT): An ecosystem of append only and tamper-proof logs, monitors and auditors to oversee the issuance of digital certificates issued by certificate authorities (CAs).

DYNAMIC HOST CONFIGURATION PROTOCOL (DHCP): A network management protocol used by network operators to automatically assign IP address identity and automate network configuration.

DOMAIN NAME SYSTEM (DNS): A hierarchical distributed system that behaves as the Internet phone book – translating human memorable names into machine readable resources such as IP addresses, or cryptographic key material among others.

DNS SECURITY EXTENSIONS (DNSSEC): A set of standardized extensions to DNS using digital signatures based on public key cryptography to provide guarantees of response authenticity and integrity.

DNS OVER HTTPS (DOH): A DNS protocol where the clients and the DNS resolver establish a secure TLS connection and HTTP messages to exchange queries and their associated responses.

DNS OVER TLS (DOT): A DNS protocol where clients and the DNS resolver establish a secure TLS connection over which serialized DNS queries and responses are sent.

DELEGATION SIGNER (DS): A specific DNS resource record containing the digest of the DNSSEC key signing keys establishing a chain of trust between the current zone and its parent zone.

END-TO-END ENCRYPTION (E2EE): A secure and private communication method that encrypts messages between senders and receivers preventing third parties including the communications provider from observing the communications.

EVOLVED PACKET CORE (EPC): The core network architecture of a 4G LTE network containing various components responsible for critical control and data plane communications. This is maintained and run by cellular network operators and contains the MME, HSS, PGW, SGW, PCRF and other network functions.

EFFECTIVE TLD (ETLD): A domain under which domains can be registered by a single organization.

FULLY QUALIFIED DOMAIN NAME (FQDN): The complete domain name of an Internet host.

GLOBAL SYSTEM FOR MOBILE COMMUNICATIONS ASSOCIATION (GSMA): An organization representing the interests and unifying mobile network operators worldwide creating the global cellular ecosystem.

HOME SUBSCRIBER SERVER (HSS): The main subscriber database containing information about the subscriber's SIM identifiers; associated access policy, billing information and other metadata corresponding with geographic location. The HSS also generates the authentication vectors used for SIM authentication.

HYPERTEXT TRANSFER PROTOCOL (HTTP): An application layer protocol designed to transfer information between devices enabling users access to resources on the Internet through a request-response process.

IP MULTIMEDIA SUBSYSTEM (IMS): A backward compatible extension to cellular network cores providing multimedia services such as voice, video, text, and data over IP networks.

INTERNET PROTOCOL (IP): Standards for addressing and routing Internet data.

INTERNET SERVICE PROVIDER (ISP): Business entity providing Internet connectivity to users.

MOBILITY MANAGEMENT ENTITY (MME): A key component of the EPC providing session management during user mobility and supports various authentication, and roaming functions.

MULTIMEDIA MESSAGING SERVICE (MMS): A multimedia messaging service allowing cellular subscribers to send media (audio, video, photos), and larger messages than supported by SMS as a single message.

NAME SERVER (NS): An authoritative DNS server responsible for managing the DNS RRs associated with a zone.

ONE TIME PASSWORD (OTP): A temporary, unique, dynamic password that is valid for a session – typically sent to a mobile device through a cellular carrier used in multi-factor authentication across the Internet.

POLICY AND CHARGING RULES FUNCTIONS (PCRF): A network function that is responsible for real time actions such as policy enforcement, and measurement of data flows responsible for billing.

PACKET GATEWAY (PGW): Core components enabling cellular network cores to connect to external IP based networks such as the Internet.

PUBLIC KEY INFRASTRUCTURE (PKI): Publicly maintained and accessible infrastructure required to create, manage, distribute, use, store and revoke digital certificates or public keys associated with identities – typically hostnames, or other identifiers.

RADIO ACCESS NETWORK (RAN): A key component of the cellular network architecture responsible for enabling radio based connectivity of user devices to the cellular network infrastructure.

RESOURCE RECORD (RR): A building block of the DNS containing information about the name and its associated value for a given resource type such as IPv4, IPv6 addresses, mail server information etc..,

SERVING GATEWAY (SGW): Critical network functions that route packets between the radio base stations to the packet gateways.

SUBSCRIBER IDENTITY MODULE (SIM): A user identity issued by the carrier network containing a cryptographic symmetric key enabling authentication to the operator network and data.

SHORT MESSAGING SERVICE (SMS): A backward compatible text messaging service allowing users to send messages between mobile devices through their cellular carrier networks.

SIGNALING SYSTEM 7 (SS7): A critical cellular control-plane protocol used for signaling between carrier networks and across a public switched telephone network establishing call and message routing and control.

TRANSMISSION CONTROL PROTOCOL (TCP): A communications standard protocol that enables networking and message exchange between Internet devices.

TOP LEVEL DOMAIN (TLD): Domains in the DNS hierarchy that immediately follow the root level eg. com., org. etc..,

TRANSPORT LAYER SECURITY (TLS): A standardized cryptographic protocol that protects communications between two end points in an Internet network providing authenticity, confidentiality and integrity guarantees for the communications.

UNIFIED DATA MANAGEMENT (UDM): A 5G core service that manages access to UDR splitting access control responsibility from a traditional LTE HSS.

USER DATAGRAM PROTOCOL (UDP): A connectionless communication protocol allowing applications to establish low latency connections while tolerating packet losess – useful for time sensitive applications such as gaming, video streaming and DNS.

UNIFIED DATA REPOSITORY (UDR): A 5GC component similar to the HSS in LTE networks but solely responsible for maintaining subscriber profiles.

USER EQUIPMENT (UE): A commercial off the shelf user mobile device.

VERIFIABLE KEY DIRECTORY (VKD): A public key directory leveraging authenticated data structures providing integrity and authenticity guarantees for the information stored in the directory.

VIRTUAL PRIVATE NETWORK (VPN): A private network created between two or more independent different networks allowing them to act as a single networked entity and optionally securing data packets.

VERIFIABLE RANDOM FUNCTION (VRF): A cryptographic function that takes a series of inputs and produces a pseudorandom output and proof of authenticity that can be verified by those with the associated public key.

Dedication

To my parents – Jyothi and Subramanyam,

for their constant love, support, encouragement, and countless sacrifices.

Chapter 1

Introduction

Our reliance on the Internet has made it a critical part of today's world and is significantly shaping our society. Today's Internet infrastructure is woven into society and powers key services such as allowing users to interact with various digital services, exercise free speech, make financial transactions with ease, gain access to important information, communicate securely, and arguably contributes positively to economy [72]. The COVID-19 pandemic changed how users interact with the Internet with many businesses embracing digital services, and communications moving fully online. Suddenly, the Internet changed from a 'good to have' to something that is critical to our daily lives impacting the way we work, socialize, create, share, and consume content and access services from around the world.

Recent reports from the Global System for Mobile Communications Association (GSMA) indicate that despite the increase in availability, affordability of Internet services across the world, and its vital role in society, the Internet usage gap remains as of 2022 at 3.2 billion people [142]. Unconnected populations are at a greater risk of exclusion as more and more services transform to be completely online. It is therefore extremely important to address the challenges of connectivity and identify mechanisms to provide secure access to the Internet.

The rapid increase in the availability of affordable Internet across the world, and reduction in the percentage of unconnected users today is fueled through extensive investments in network infrastructure by nation scale cellular networks [142]. Despite the intent to connect the unconnected, the economics make it extremely difficult for a small number of national scale cellular network operators to deploy expensive network infrastructure, lease towers for setting up base stations, lay cables and optic fibers underground or over electrical transmission lines [147]. This brings to light the need for a collective effort to address the challenges of connectivity that remain and bring over a billion people online. Various network connectivity solutions such as Fixed-wireless Internet Service Providers (WISPs), city scale mesh networks [67, 143, 227], and Community Cellular Networks (CCNs) have been proposed to address the challenges of connectivity and provide affordable Internet access – especially in areas where it is difficult to obtain the services of nation scale cellular networks [144, 147, 284]. Modern protocol standards in 5G networks enable host neutral architectures where connectivity services could be provided regardless of the mobile carrier networks of the users – allowing these carriers to extend network connectivity on demand.

While the rapid growth in the adoption of the Internet is fueled by large nation scale cellular operators, these operators have traditionally been few and create an oligopoly [22, 290, 312, 339]. It is today increasingly economically efficient to provide access to Internet in dense population zones with deployment of cellular infrastructure than through wired broadband connectivity. The access to Internet services through cellular operators transform the cellular carriers into large Internet Service Providers (ISPs). Smaller organizations and communities are therefore relying on growing popularity of 5G – enabling host neutral architectures to deploy and manage private 5G/LTE networks using unlicensed spectrum [350] such as the Citizen Broadband Radio Service (CBRS) to provide users access to Internet [36, 221] at an even more affordable cost while also empowering the local community to adapt the network as needed [3, 171, 175]. Users of these networks interact with and trust various hidden services enabling their Internet connectivity.

While the deployments of private 5G/LTE networks across the world address the growing challenges of Internet connectivity and affordability, their usage is significantly limited within a specific geographic area compared to nation scale cellular networks because of their lack of ability to enable user mobility. Due to the nature of mobile communications, users of cellular networks travel physically between geographic locations with their devices and in small scale private 5G/LTE core deployments, lose access and connectivity when outside the range of the network. However, the issue of providing users access to the network services during their mobility is addressed easily and efficiently through large nation scale cellular networks because of the centralized nature of the network and the ability to have total visibility into its network services.

Prior works focused on improving Internet access have focused on addressing the challenges of *access* and scaling the services to provide equitable access to network services. While the mechanisms have resulted in the development and growth of private 5G/LTE networks, the nature of today's cellular infrastructure that provides users access to the Internet present two key challenges. First, the existing architecture of cellular network deployments by nation scale operators resulted in heavily trusted and architecturally centralized core networks enabling a few organizations to control access to, and the type of services offered to its users. Second, while the heavy centralization enables economies of scale, and reduces security challenges involved in operations since they happen within a trusted control zone, it opens up tremendous potential for abuse, enables potential censorship, and adversely impacts user privacy.

The origins of today's Internet on the other hand put heavy emphasis on decentralization of the resources while focusing on the interoperability between various independent networks [82]. However, the access to such distributed network of networks today is enabled through a few access providers. Once users gain access to the Internet through these cellular networks or Internet service providers (ISPs), they interact with various infrastructure services hosted and run within the network perimeter of the network provider. These hidden services allow users to successfully

and securely navigate on the Internet. With the rapid growth of cloud computing, and the widespread usage of Content Delivery Networks (CDNs) over the last decade, most Internet traffic today flows through the networks of a select few large companies rather than its decentralized humble beginnings [237, 313]. As a result, the architecture of the Internet is constantly changing and being reshaped to meet today's computing needs. This co-location of services, and content provided by CDNs and cloud computing resulted in today's Internet traffic being routed through a small set of companies with global scale and footprint. These organizations put themselves in a highly privileged position because of the implicit trust placed in their services and are privy to an incredible amount of private information of their users.

Another often overlooked part of today's network connectivity is the interdependence that cellular and Internet networks have on each other. Cellular network providers essentially act as the guardians of user's digital identity allowing users to enroll with various Internet based services. As a result, the usage of carrier driven identity allows these service providers to validate and identify their users preventing sybil attacks where the services could be overwhelmed due to fake identities on their platforms. While the boundaries between cellular and Internet networks intersect, and have many similarities such as their packet switched nature of communications over which various applications (messaging, voice etc.,) are built, they're equally dissimilar in the matter of communication security, privacy and the implicit trust placed in the infrastructure. For example, due to their open nature, the public key infrastructure (PKI) enabling authenticated communications on the Internet today is auditable and extensively monitored. However, such PKI infrastructure in cellular networks are privately maintained, and not publicly auditable raising concerns about their correctness and validity – requiring users to implicitly trust them with the correctness of their own identity. The lack of the same security and privacy guarantees in cellular networks which are provided by Internet networks despite their growing interdependence puts user's privacy at risk.

In this thesis, I posit that future connectivity solutions will witness a rise in the deployment of private cellular networks which provide Internet and network services. During natural events such as human mobility, it is therefore increasingly important to enable cooperation between smaller scale individual network cores and allow users from different networks to seamlessly roam between unrelated private cellular network cores. Thus, keeping in mind the growing demand for computing and the increasing co-location of services, both at the *access* layer to provide users access to the Internet and at the *usage* layer where users communicate with various resources on the Internet, I argue that it is difficult to achieve today's scale, performance, and ease of access provided by networks today and therefore it is important to embrace it and identify practical and performant mechanisms to protect user privacy and security, and empower service providers to transparently prove their operational correctness and honesty to their users. Finally, I challenge the existing notion of implicit trust in networking infrastructure and argue that it can be reduced by improving the transparency of interactions – additionally providing benefits of provable operational correctness to network operators.

1.1 Thesis Statement

The usage and adoption of cellular and Internet networks for crucial societal functions is increasing and any infrastructure failures cause significant disruption to the population relying on it – making this infrastructure critical societal computing infrastructure. Such infrastructure today is operated, and maintained by a small set of organizations who we increasingly trust with an immense amount of user data. This thesis challenges the notion of such *implicit* trust, advocates for the need for **verifying** operational correctness of, and interactions with this infrastructure, while **improving** user privacy without adversely affecting **performance** – focusing on its immediate **practicality**.

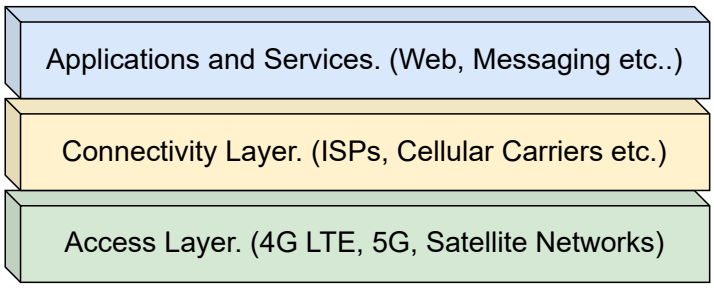


Figure 1.1: Three Functional Pieces of Today's Internet

1.2 Thesis Contributions

My thesis work makes an argument that the Internet today could be seen as three individual functional pieces – shown in Figure 1.1.

1. The **Access** layer allows users to gain access to the Internet through the use of technologies such as 4G LTE, 5G, satellite, fixed wireless, wired fiber or cable networks. The organizations providing these services are (1) typically a few, (2) operate at national scale, (3) are implicitly trusted to provide connectivity, (4) have complete control over the network traffic, and (5) visibility into user's network access patterns.

2. The **Connectivity** layer enables organizations providing access layer services to interconnect with other networks. This layer provides additional hidden network services such as DNS, routing, gateways and interacts with various CDN and cloud providers which are critical – allowing the users to access various Internet applications and services globally.

3. The **Application and Services** layer consist of the individual end user applications and Internet enabled services that users connect to through their access providers. These include popular Internet enabled instant messaging, email and communication services, and web applications among many others.

At each layer, I ask and attempt to address the following primary research questions:

(a) What is the nature of trust between the users and the current layer?

(b) What are the security and privacy challenges users face while interacting with this layer?

(c) How can we improve user security and privacy?

(d) What are the (un)intended consequences of improving user security and privacy?

Together, I hope this thesis presents an interdisciplinary perspective on the security and privacy challenges for users on the Internet. I hope my work highlights the need for identifying hidden and implicitly trusted infrastructure. My work argues for implementing technical mechanisms addressing the challenges of trust which are easy to understand, and deploy globally.

I use a range of methods in my work drawing on existing legal challenges, and relying extensively on large scale Internet measurements either as a need finding exercise or to showcase the current state of security and privacy of various network infrastructure components on the Internet. These exercises motivate the design and development of new protocols resulting in secure systems development, its performance evaluation, and eventual deployment.

1.3 Published Materials and Contributions

This thesis contains parts which have been published in peer-reviewed settings and presented during various recorded meetings of Internet standards bodies. Some of these works are in-part a reprint of the original material where the dissertation author was the primary investigator and author of the published works, some of which are done in partnership with Cloudflare. Additionally, it contains information from works that are pending publication.

1. *Sudheesh Singanamalla, Esther Jang, Nick Durand, Matthew Johnson, Spencer Sevilla, Kurtis Heimerl.* 2020. dAuth - Decentralizing LTE Authentication and Roaming. Presented at

the Decentralized Internet Infrastructure Research Group (IRTF DINRG) interim meeting 2020. https://datatracker.ietf.org/meeting/interim-2020-dinrg-01/materials/slides-interim-2020-dinrg-01-sessa-dauth-decentralizing-lte-authentication-and-roaming

2. *Sudheesh Singanamalla, Esther Han Beol Jang, Richard Anderson, Tadayoshi Kohno, and Kurtis Heimerl.* 2020. Accept the Risk and Continue: Measuring the Long Tail of Government https Adoption. In Proceedings of the ACM Internet Measurement Conference (IMC '20). Association for Computing Machinery, New York, NY, USA, 577–597. https://doi.org/10.1145/3419394.3423645

3. *Sudheesh Singanamalla, Suphanat Chunhapanya, Jonathan Hoyland, Marek Vavruša, Tanya Verma, Peter Wu, Marwan Fayed, Kurtis Heimerl, Nick Sullivan, Christopher Wood.* 2021. Oblivious DNS over HTTPS (ODoH): A Practical Privacy Enhancement to DNS. In Proceedings on Privacy Enhancing Technologies Symposium (PoPETS'21). Issue 4. Pages 575-592. https://doi.org/10.2478/popets-2021-0085

4. *Sudheesh Singanamalla, Muhammad Talha Paracha, Suleman Ahmad, Jonathan Hoyland, Luke Valenta, Yevgen Safronov, Peter Wu, Andrew Galloni, Kurtis Heimerl, Nick Sullivan, Christopher A. Wood, and Marwan Fayed.* 2022. Respect the ORIGIN! a best-case evaluation of connection coalescing in the wild. In Proceedings of the 22nd ACM Internet Measurement Conference (IMC '22). Association for Computing Machinery, New York, NY, USA, 664–678. https://doi.org/10.1145/3517745.3561453

5. *Sudheesh Singanamalla, Richard Anderson, Kurtis Heimerl.* 2024. Poster: Retrofit – Enabling Interoperable E2EE Communication Through 6G Cellular Architectures. Poster Presentation at 2024 Network and Distributed System Security (NDSS) Symposium (NDSS'24). https://www.ndss-symposium.org/wp-content/uploads/ndss24-posters-41.pdf

6. *Evan Lam, Richard Anderson, Kurtis Heimerl, Yurie Ito, Jonathan Joseph de Koning, Adam M. Lange, Jarrod O'Malley, Adam Shostack, Arastoo Taslim, Sudheesh Singanamalla.* 2024. Poster: On the (In)Security of Government Web and Mail Infrastructure. Poster Presentation at 2024 Network and Distributed System Security (NDSS) Symposium (NDSS'24). https://www.ndss-symposium.org/wp-content/uploads/ndss24-posters-42.pdf

7. *Sudheesh Singanamalla, Ben Weintraub, Christian Elmerot, Marwan Fayed, Kurtis Heimerl, Cristina Nita-Rotaru, Thibault Meunier.* 2025. DNSSEC For The People: Bringing DNS Resolution Transparency to Clients. *(Under Submission)*

1.4 Thesis Outline

This thesis presents the required background and closely related work in Chapter 2. Chapter 3 details how users place implicit trust in today's cellular and Internet infrastructure across all three layers of connectivity and brings to light the security and privacy challenges due to the hidden infrastructure which also apply to widely used and ostensibly secure and private networks. Chapter 4 focuses on the access layer, particularly the cellular networks and presents an authentication system which enables decentralized cellular networks allowing users to reduce implicit trust placed in centralized cellular networks for identity, authentication and service. Chapter 5 brings to light the need for and challenges in improving the security of a critical but often overlooked part of the Internet – its tail. Chapter 6 focus on the hidden and implicitly trusted infrastructure in the connectivity layer – particularly DNS and the certificate authority (CA) ecosystem and highlights various opportunities to improve security and privacy. Despite improving security and privacy across the access and connectivity layers, chapter 7 identifies core applications and protocols whose operational correctness is critical and heavily trusted. It presents highly practical mechanisms in which the correctness of these systems could be audited and advocates for transparency. Finally chapter 8 presents the concluding thoughts and opportunities for future work.

Chapter 2

Background & Related Work

I situate my work in the context of cellular networks due to their widespread adoption as the low-cost, scalable and reliable method to provide Internet connectivity to their subscribers. Specifically, the host-neutral architecture of 5G, LTE networks enabled an increasing number of communities across the world to deploy cellular networks that provide low cost Internet access to those areas where commercial cellular network connectivity is not deployed due to lack of economic viability, or to those users who cannot afford cellular network subscriptions provided by nation scale operators. Community deployments of these networks are used to provide connectivity services to the most vulnerable populations whose privacy and security need to be preserved during their online interactions. Typically, these deployments tend to be localized to a specific geographic area, have lacking security and privacy practices around infrastructure maintenance and operations putting their users at risk.

2.1 Core Cellular Network Architecture

From a birds eye view, the cellular network architecture contains three different components: (1) A network *core* managed by the network operators, (2) the radio access network layer (RAN), and (3) the user equipment or devices (UE). Each of the individual components of the cellular networks

contains a *user*, and *control* plane. The *control* plane carries instructional/signaling messages between the various components of the network and is used to manage the operations and access to services provided by the network. The *user* plane transmits the data packets from user devices destined to various services offered by the network provider or the Internet.

The UE is typically a mobile cellular device supporting multiple cellular protocols (GSM, CDMA, etc..,) and consists of a network operator issued Universal Integrated Circuit Card (UICC), which is commonly referred to in cellular networks as the Subscriber Identity Module (SIM) card. These physical modules/cards store very specific data such as the user's phone number, authentication state and security keys information. The wireless communications between the UE and the core network are handled by the RAN layer and are facilitated by the deployment of physical radio base stations. In the 4G LTE networks, these base stations are referred to as evolved base stations (eNB) [255], and in 5G are knows as Next Generation Node B (gNBs) [319]. While the terminology of the infrastructure changes between different generations of networks, they are responsible to maintain a few key roles such as controlling and managing the connectivity state and radio layer communication security of multiple devices. The base stations encode digital communications intended for a specific UE through digital signal processing techniques and control the connectivity operations of the mobile devices connected to it.

In typical deployments, the communications between the base stations and the core network are established either through dedicated private fiber wired links, or routed over the network of an existing Internet service provider in the area. A cellular core network, also known as an Enhanced Packet Core (EPC) in 4G LTE, and 5G Core (5GC) in 5G networks are run by a centralized network operator within their data centers for nation scale cellular networks, or are run individually by operators of private 5G or community cellular networks. The core network adopts a service driven architecture and is composed of various services offering a functional split.

In Figure 2.1, we show a typical backward compatible co-located 4G and 5G architecture

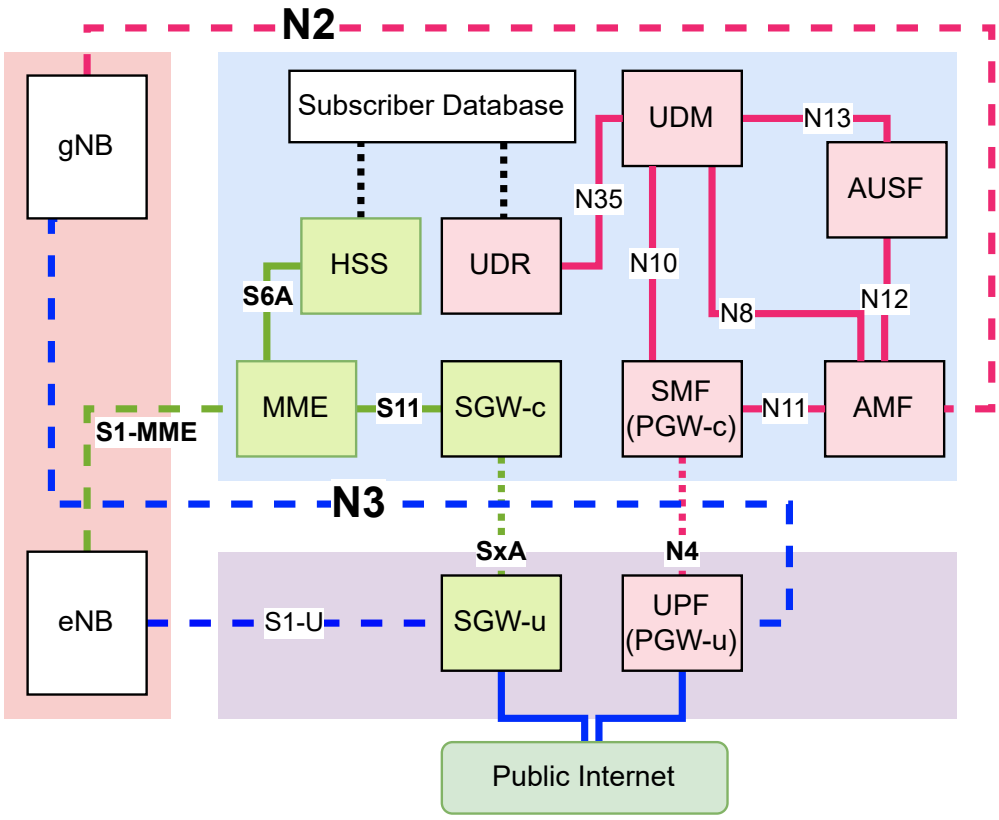


Figure 2.1: Architecture of a 5G Network Core (5GC) with backward compatible 4G LTE capability.

Figure shows a 5G Core network setup with the necessary components (light pink) supporting backward compatibility with 4G LTE EPC components (light green). The solid green and pink lines indicate the various control plane communication protocols used by 4G and 5G components. The dotted black line connecting the HSS and the UDR with the subscriber databases perform the required database lookups and validations. The dashed blue lines indicate user plane communications between the base station and a carrier configured user software gateways enabling the users to communicate with the public Internet.

with the essential components required to run an individual core network and is typically the configuration used by cellular networks supporting user devices capable of both 4G LTE and 5G. At a minimum, the EPC in 4G LTE consists of the Home Subscriber Server (HSS), Mobility Management Entity (MME), and a Packet/Serving Gateway (PGW, SGW). The HSS contains services which interact with the subscriber database where all the information corresponding

to the network's subscribers including the keys needed for authentication reside. The MME is the critical signaling node of the EPC and is responsible for interacting with other services and managing the access control of the UEs and various radio resource management functions. The packet gateway communicates with the public Internet and routes user data packets to the Internet granting users access to the Internet.

The evolution of 5G identified some of the key challenges in 4G networks and decomposed the modules into various smaller services with dedicated roles. The MME and HSS in 4G core network are decomposed in the 5G networks into an Access and Mobility Function service (AMF), the Unified Data Repository (UDR) services which interact with the databases. In an effort to make more services stateless allowing operators to scale individual services, the Unified Data Management (UDM) services interact with UDR to obtain necessary information about a network subscriber or the state of the subscriber. To improve security, the authentication and validation mechanisms were further decoupled in 5G cores giving rise to the Authentication Server Function (AUSF) which facilitates the cryptographic authentication operations and key maintenance.

In addition to the set of modules needed to establish basic cellular network functionality, network operators deploy additional infrastructure such as DNS resolvers, Policy and Charging Rules Functions (PCRF) mechanisms [217] and network slicing [16], both at the radio layer and the packet user plane layers to provide variable quality of service and appropriate billing to their subscribers [356]. Despite the advances made in cellular networking protocols, all protocol variants attempt to be backward compatible and allow the usage of Short Messaging Service (SMS) using the SMS Functions (SMSF) enabled through SMS over NAS (Non-Access Stratum), or through SMS over IP Multimedia subsystem (IMS) [77]. Cellular network providers can additionally provide these services to their subscribers.

2.2 Internet Infrastructure, and Secure Transports

Domain Name System (DNS): DNS is a system designed to map human-readable domain names to various resources such as IP addresses and texts, among others [233]. It has an interface much like a distributed key-value store. These domain-resource mappings are stored in *resource records* (RRs). DNS stores these mappings in distributed *zones* arranged hierarchically in a tree structure. At the root of the tree lies the *root zone*. The root zone is maintained by the Internet Assigned Numbers Authority (IANA). The domain name of the root zone is denoted as a single period (“.”). Under the root zone are the zones of the top-level domains (TLDs)—these include the suffixes any Internet user will recognize (e.g., .com, .org, .net). in addition to suffixes for most countries (e.g., .uk, .it, .au)—called *country code TLDs* (ccTLDs). The hierarchy continues below the TLD zones, but these zones are not maintained by any central or governmental authority. These are typically made available to purchase through domain registrars. The complete domain name, including the trailing ‘.’ for the root, is called a Fully Qualified Domain Name (FQDN) (eg. example.com.).

Each zone contains a set of *nameservers*. Nameservers are typically deployed in a replicated configuration to increase reliability and ensure high availability. Each nameserver stores either the resource records of the domains within its zone, or a delegation to another set of NS record mappings to nameservers in a child zone. For example, consider a query requesting the NS information for example.com.. The query requests the NS information from the com. zone which are commonly referred to as the TLD nameservers, which results in an answer containing the *authoritative* nameserver for the example.com. domain. Recursively, the information about TLD nameservers, can be obtained from the root zone (‘.’). This authoritative nameserver might also manage the resource records for its children, i.e., the www. subdomain. Authoritative nameservers are the source of truth for the resource records.

When a client wishes to perform a DNS resolution, they can do so by traversing the DNS zone

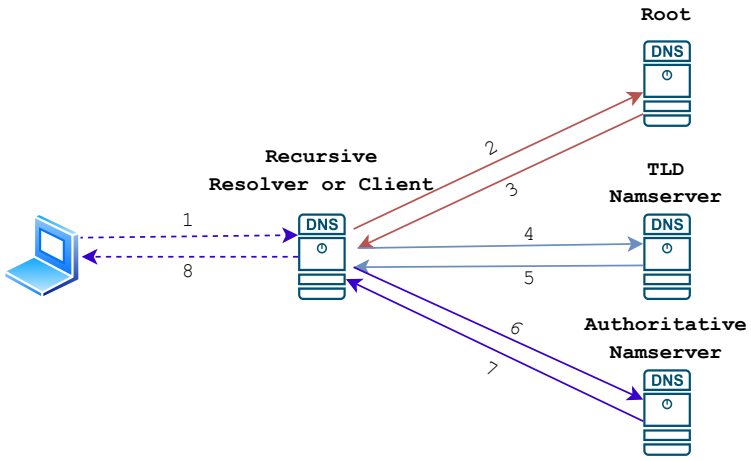


Figure 2.2: DNS Recursive resolvers perform the necessary resolutions recursively (step 2-7) on behalf of the client and return the responses to the client. The client perceives 1 RTT latency for the response which includes the time to perform the required resolutions.

hierarchy starting at the root level. We illustrate this process in Figure 2.2 (requests labeled 2–7). The entity performing the resolutions is called a *resolver*, and resolvers running on a client are called *stub* resolvers. For each zone, the client sends a DNS Question for an associated resource record type (RRType) to the zone's nameserver and receives a DNS Answer. The response will either be a set of all RRs of the requested type, or contain nameserver information for the next zone which might be able to answer the query, or one of the "record not found" types. The nameserver can also provide *glue records* – RRs that were not specifically requested, but will be eventually necessary for resolution; the nameserver can provide these if it knows that the client will need to request them in future DNS queries. In general, the resolution process starts from the root, moves to the TLDs, and then traverses zero or more second-level domains and subdomains.

The process of resolving a query to an answer using the DNS described above is inefficient when repeatedly performed by clients wanting to visit websites and interact with various services on the Internet. A straightforward optimization is to cache the responses from the root and the TLD resolvers until a specific validity time, known as Time To Live (TTL) provided in the

resolver responses. While this prevents repeated queries, it increases the burden and complexity of the client stub resolvers who would need to maintain these caches. Better than individual client caches, are large shared *resolver* caches deployed at the network perimeter, either by ISPs or cellular carriers which are used by their subscribers. Network operators use standardized configuration packets (through Dynamic Host Configuration Protocol – DHCP [113]) when users join the network to configure the information of the resolver. This allows clients to delegate the resolution and the maintenance of the caches to the DNS services provided by the service provider and allows clients to send a query and receive a response in a single request-response round trip interaction shown by step 1 (request), and step 8 (response) in Figure 2.2.

DNS Security Extensions (DNSSEC): DNS communications were initially designed to be plaintext resulting in network adversaries between the clients and the resolvers performing active attacks tampering or modifying the requests or responses. DNS communications are also monitored by passive non-tampering adversaries to understand user access patterns affecting their browsing privacy. DNS Security Extensions (DNSSEC) aim to protect against network adversaries modifying or forging responses from authoritative nameservers maintaining response integrity through cryptographic signatures. DNSSEC maintains the *integrity* and *authenticity* of DNS responses from authoritative nameservers using cryptographic digital signatures and relies on public key cryptography. Successful DNSSEC usage relies heavily on the hierarchical nature of DNS which allows the maintenance of a decentralized public key infrastructure (PKI). Authoritative nameservers append cryptographic signatures to the collection of resource records (RRSet) which are returned to the requesting client. Therefore, the response to the client query requesting a DNSSEC validated response includes the set of RRs and an associated signature ($\mathrm{RR{+}RRSIG}$).

Each DNSSEC-enabled zone manages its own key pairs and publishes the public parts of the Key- and Zone-signing keys (KSK and ZSK) as DNSKEY records which in-turn is signed by the

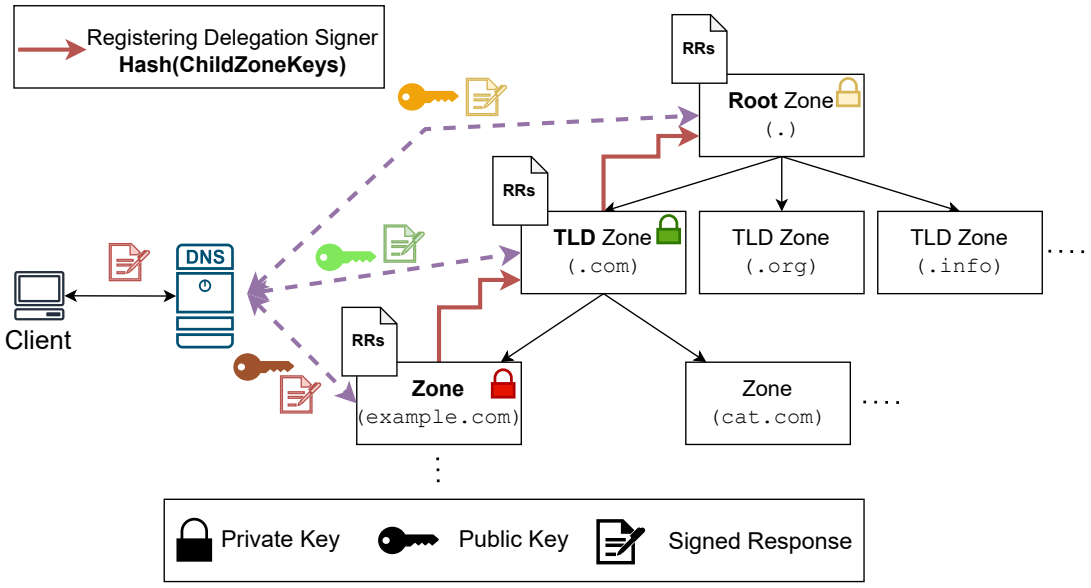


Figure 2.3: Traversing the DNS hierarchy to resolve a DNSSEC query Q = `example.com`(A)

KSK, while the remaining RRs associated with the zone are signed using the ZSK. To avoid a solipsistic dilemma, the parent zone must attest to the correctness of the KSK – done in the form of a *delegation signer* (DS) record containing a cryptographic hash of the KSK validated and signed by the parent zone's nameservers.

As shown in Figure 2.3, a client query to a validating resolver results in the resolver requesting signed DNSKEY's, and associated DS's from authoritative nameservers in the DNS hierarchy as shown by the colored keys and signed resource record responses along the dashed purple arrows. Each response is individually validated by the resolver and the cryptographic signatures from the response (red signed file icon) obtained from the authoritative nameserver for the requested zone is sent as the final answer to the client query. The client in this approach delegates the responsibility to validate the content of the DNS responses to the recursive resolver without which the client performs all actions of the resolver and validates the individual responses.

HTTPS, Certificates, and the Web: https is an encrypted data transfer protocol between a web browser client and a web server providing a secure version of the older http protocol. https uses Transport Layer Security (TLS), a successor to the now deprecated Secure Sockets Layer (SSL) protocol, to establish secure communication using asymmetric key cryptography. A TLS handshake starts the process of establishing a secure connection to an endpoint once an underlying transport protocol connection (typically Transmission Control Protocol (TCP)) has been established. Web browser clients resolve the host IP addresses for a given Internet resource via DNS resolvers and establish a TCP connection with the host IP addresses on a well known port (80 / 443). The TLS handshake begins with the client and server negotiating the TLS version and cipher suite to use. The client validates the certificate provided by the server, then generates a *premaster secret* which is encrypted with the server's public key. This is used to establish a shared session key, enabling a secure encrypted communication channel [92].

TLS/SSL certificates are specific files hosted by web servers containing the host's public key along with identity information, such as the domain name they wish to use and the name of the organization. These certificates are (per best practice) cryptographically signed (attested) and issued for a fixed duration by a trusted certificate authority CAs. CAs previously voted to limit certificate lifetimes to 825 days [69], and further reduced lifetimes to one year starting September 2020 [2, 70], eventually reducing the lifetimes to 90 days [307]. However, it is also possible to create a *self-signed certificate* which is cryptographically valid but not attested by a CA – preventing browsers from accessing the resources by presenting warnings about the interaction and only performing the connection when explicitly bypassed.

If configured correctly, https allows communication to remain confidential and non-tamperable, providing an authenticated medium between client and server with the assurance that communications are only being received and read by the intended recipient. Popular projects like Let's Encrypt, a US non-profit, have made it possible for website operators to add https support for

absolutely no fee [8]. Cloud providers like Azure, Google Cloud, along with Content Delivery Network (CDN) providers like Cloudflare and Akamai, have similarly made it easy to integrate https by intercepting and handling the requests in a secure, easy to configure manner [119]. http, without https, due to its plaintext nature, enables *man-in-the-middle* (MITM) attacks wherein the adversary can eavesdrop, secretly alter, and relay communications between two parties.

Expert attackers, malicious governments, or Internet Service Providers could proxy requests and show modified content to the user, steal their information, or use it for surveillance. Such attacks have been thoroughly studied and publicly documented by cybersecurity companies [6] and organizations such as the Open Web Application Security Project (OWASP) [94]. The lack of a matching root certificate during validation of a certificate chain results in an error indicating undetected local issuer certificate – typically preventing users from navigating to the resource unless browser security policies are explicitly disabled [259].

Certificate Authorities (CA): Certificate Authorities (CAs) are trusted third parties whose core responsibility is to issue SSL/TLS certificates [8]. CAs and their certificates are treated as trust anchors and shipped by default by software providers (usually with browsers or operating systems) such as Microsoft, Google, Apple, and Mozilla [24, 229, 240]. The list of default trusted root CAs can differ between browsers and tools [311]. Our analysis of the trust stores show that Apple includes 174 default root trusted certificates, while Microsoft [229] includes 402 default root certificates. The Mozilla NSS [240] trusted certificate store consists of 152 default root trusted certificates. NSS trusts 52 individual root CA owners, while Microsoft and Apple trust 133 and 69 root CA owners respectively. Any valid intermediate CA must be authorized as a CA [51]. Therefore, a weak CA in a certificate trust chain is a weak link for website security, exemplified by attacks on DigiNotar [270] and Comodo [18, 23] resulting in increased efforts focused on improving transparency.

The inclusion of a CA in the trusted roots require strict issuance compliance and periodic audits [242, 306]. A certificate issued by a CA binds the public key of the web host to the domain name and is cryptographically established by the CA signing the contents with its private key. A CA responds to a request to issue a certificate by challenging the domain host to prove its ownership. Such Domain Validated (DV) certificates are the most common type. CA-issued certificates can also include information such as organization names, postal address, or an administrator email address. These Extended Validated (EV) certificates are rigorously validated by the CA before issuance and are intended to make phishing attacks with valid certificates harder. However, it was still possible for a malicious attacker to register a company with the same name in a different physical address and request an EV certificate. EV certificates are generally expensive, with a fee for issuance. They have been widely adopted by large Internet companies, payment gateways, and banks providing online services. However, their popularity has reduced due to concerns about their effectiveness and the move by major browsers to avoid distinguishing visually between EV and DV certificates in their respective browser interfaces [318].

2.3 Addressing Global Connectivity Challenges

The typical deployment of a community cellular network or private 5G networks aimed to address the long tail of Internet connectivity are done as small scale independent cellular operators operating and offering services to users within a specific geographical region. The network deployments typically consist of an edge network core running necessary services and connected to a single base station [147].

2.3.1 Increasing Deployment of Private Cellular Networks

Work on rural community networks has historically ranged from improvements in long-range wireless protocols [38, 264] to mesh and distributed architectures and deployments [11, 35, 168,

256]. Rural network deployments additionally encounter and address the challenges of maintaining backhaul and power [325]; Hasan *et al.* make the case for graceful degradation of service in the face of failure rather than a hard system cutoff [144]. Community cellular networks (CCNs) have been explored by Heimerl *et al.* [146–148] as a way to offer wider-area coverage than WiFi based networks, provide basic telephony as well as IP-based services, allowing communities to focus maintenance on one site of failure as opposed to a community-wide mesh [173]. 2G [40, 144, 148, 284], 3G [220], and 4G [173] community networks have been deployed either as isolated single-instance networks in a region, or have relied on a central network management server (accessible via the backhaul link) to coordinate network state.

Due to their remoteness, it is typically important for such networks to be able to operate in a disconnected fashion to serve users' local traffic, even when the backhaul link to the outside world is down [144, 309]. These networks are often connected to the rest of the Internet via a low-bandwidth or unstable backhaul connection, for example VSAT satellite or long distance WiFi links with only 1-3 Mbit/s throughput, variable latency, or low uptime due to weather or power disruptions [144]. The local presence of the "edge" network core is able to facilitate this disconnected operation.

Even if there are multiple related communities or villages clustered together in a given region as typical in many Indigenous tribal and other rural settings around the world, each community may want to have its own network core and backhaul connection for political reasons as well as to reduce shared points of failure and network complexity.

2.3.2 Efforts at Distributing Cellular Core Network Functionality

Prior research efforts such as CellBricks [214] present an architecture lowering the barriers for new cellular operators to provide their users access on-demand from any available cellular operator by moving the support for mobility and management out of the network into end hosts thus

requiring changes to the user devices. *dLTE* [173] is a proposed decentralized LTE architecture that provides data only services rather than traditional telephony; it proposes removing network level authentication and user management, rely on over the top security, authentication, and billing at the application layer.

Jover and Lackey propose *dHSS*, a distributed decentralized Home Subscriber Server (HSS) implementation for LTE which is a blockchain-based public key infrastructure (PKI) model instead of the symmetric key model of today's HSS. However, they rely on asymmetric cryptography at the network layer which is incompatible with current devices [177]. The usage of individual symmetric keys per subscriber is baked into the older 2G, 3G, 4G and 5G specifications for legacy reasons, one argument for this being the decreased ramifications of having a single subscriber's key leaked, as opposed to a network wide private key. It'll likely be over a decade before 6G potentially incorporates changes. Alternatively, Johnson *et al.* implement dAuth, a federated mechanism to provide subscriber authentication in cellular networks without relying on a single cellular network provider using cryptographic secret sharing mechanisms [174].

2.4 Cryptographic Transparency & Verifiable Infrastructure

Until 2013, the traditional model of certificate issuance by a trusted CA left users vulnerable in case the issuing CA is compromised. Attacks such as the 2011 breach of DigiNotar [270] and 2012 Trustwave sub-ordinate root certificate policy resulting in TLS connections being monitored [275] led to the proposal, and widespread deployment of Certificate Transparency (CT) infrastructure [203]. The proposal was later updated and includes a TLS extension that allows servers to present signed certificate timestamps and proof of inclusion of certificates in CT log infrastructure [204].

CT log infrastructure maintains publicly available append only logs which are cryptographically maintained and monitored. Today, various organizations such as Google, DigiCert, Let's Encrypt, Sectigo, Cloudflare and Trust Asia maintain publicly accessible CT Log infrastructure making up

the ecosystem enabling the success of TLS and WebPKI. CAs issue *pre-certificates* for a requested certificate from a domain owner after necessary domain validation. These pre-certificates, contain the same information that would be present in the final issued TLS certificate and are sent to one or multiple CT log servers. The CT log services issue a signed certificate timestamp (SCT) to the requesting CA and are added to the append-only logs.

Logs maintained by the CT infrastructure are maintained by using datastructures known as Merkle trees which are binary trees where each leaf node are the hashes of the individual certificates and each parent level in the tree is constructed by hashing the pair of child nodes. The log servers cryptographically sign the root of the tree creating a Signed Tree Head (STH). The SCT received by the CA is appended to the certificate to be issued and sent to the domain owner requesting the certificate. These certificates are served during the TLS handshake by the host server to the requesting browser client. The browsers validate the correctness of the certificate by traversing the certificate chain until a trusted root certificate is obtained.

The CT log infrastructure prevents the attacks such as the ones possible by TrustWave's policy of sub-ordinate certificate issues by allowing publicly run *monitor* servers that watch for suspicious certificate issuance. CT log enable transparency of the Web PKI infrastructure and attempt to bring to light any attempts at *compelled certificate creation attacks*, where government agencies might compel CAs to issue fake TLS certificates in attempts to covertly intercept and hijack secure web communications [316]. Such attacks have been identified and recorded in the past, such as the 2015 attempt by Kazakhstan to create a root certificate allowing the ability to intercept HTTPS traffic [276]. These root certificates were promptly removed from the Google and Mozilla root certificate stores [261, 329, 351]. Arguably, improved transparency and monitoring infrastructure poses significant challenges for national intelligence efforts possible via compelled certificate creation attacks due to the monetary, operational and reputational penalties for CAs when monitors or users discover such behavior.

Chapter 3

Understanding and Challenging Implicit Trust

3.1 Cellular Networks are Implicitly Trusted

The boundary between today's cellular and Internet networks blurred with most cellular networks operating as Internet service providers and establishing a packet gateway for their subscribers to connect to the global Internet network. The relatively small number of national scale cellular networks providing the access layer for subscribers makes it challenging since a small number of these providers are globally responsible and implicitly trusted for enabling user's access to the Internet. Similarly, the cellular networks are the de-facto source of digital identity for majority of users in the world due to their ability to provide unique subscriber identities bound to usable and memorable phone numbers through hardware SIM cards.

A cellular subscriber purchasing a SIM card and a subscription plan from a cellular service provider implicitly trusts the service provider to successfully authenticate the SIM and provide network services. The cellular service provider establishes trust in the subscriber by validating the SIM issuance request either through the usage of a government issued national identity in the country of operation or alternate mechanisms. In addition, the cellular network operator obtains the required *spectrum* licenses to operate within the country granting the service provider exclusive rights to use particular frequency bands which are advertised from the radio base stations.

The issuance of the SIM card by the service provider grants the device a unique SIM identifier, and includes a cryptographic symmetric key mapped to a logical phone number. Due to the symmetric key nature of the SIM card, the same key is also present in the subscriber databases operated by the cellular carrier. Any connectivity to the cellular services is provided by establishing bi-directional authentication between the service provider and the user device where the device authenticates the network and the network services authenticate the device [1].

3.1.1 Radio Layer Security and Trust

The security and privacy for users interacting with the access layer of cellular networks i.e. communication between the UE and the cellular base station depends heavily on the implementation of radio communications protocols. 2G based networks are vulnerable to false base station attacks which trick UEs into connecting and relaying messages to a malicious base station leaking their unique SIM identifier information (IMSI), and allowing the recipient fake base station to intercept, or inject traffic. The radio encryption mechanisms between the base station and the UE also uses weak encryption algorithms which are variants of the A5 encryption algorithm [274]. These algorithms were deliberately weakened for export regions allowing national intelligence organizations and law enforcement to intercept and view communications [42, 165, 250]. Despite the deployment of new protocols such as 4G LTE, and 5G which provide many security and privacy benefits, mobile devices provide backward compatibility and support communication protocols such as 2G. Mobile operating system manufacturers have only enabled the options to disable insecure 2G connectivity as recently as 2022 [336]. 2G also introduced various services which are supported by cellular carriers as value added services such as Multimedia Messaging Service (MMS) and Short Messaging Service (SMS) which are backward compatible and used even within the today's modern 5G networks. Authentication in 2G is one-sided, only allowing networks to authenticate the UE resulting in the UE trusting *any* network providing connectivity services.

The next generation of radio networks (3G) improve on the challenges posed by 2G network security allowing UEs to authenticate the network that it is attaching to, allowing the network operators to prove their legitimacy to the intended subscriber. Additionally, insecure A5/1 stream cipher based encryption algorithms were replaced by KASUMI block ciphers [180]. However, the radio layer networks were vulnerable to *rogue base station downgrade attacks* which are unlicensed radio base stations that force UEs to downgrade connectivity to the older 2G protocols. Despite improvement in the generations of radio technologies, the existence of rogue base stations attempting to downgrade the connectivity is an active threat even with modern LTE and 5G networks and likely will continue into 6G networks due to the large number of commercial products and the tensions between law enforcement's interception requirements and user security and privacy [263]. Due to the critical nature of cellular networks and the wide variety of connectivity support by devices, older protocols such as 3G continue to be operational in many countries across the world with estimated shutdown dates extending upto 2033 [95, 102].

While the protocols to secure radio communications between the base station and the UE continue to improve in the recent 4G LTE, and 5G networks, users continue to trust and rely on their cellular service providers for connectivity and communication services. UEs as a result provide information about their device such as the unique International Mobile Equipment Identity (IMEI), their physical GPS location or proximity to the connected base station and the carrier provided identity IMSI allowing carrier networks to track the location history, and communication patterns [79, 149, 162]. Cellular operators could sell, share with partners, or leak the identity and location data information when compromised by attackers [302]. Recent proposals such as the Pretty Good Phone Privacy (PGPP) attempts to address these challenges posed by permanent identifiers providing cellular network subscribers greater guarantees and reducing the need to trust their cellular operator with this information [302].

3.1.2 Core Network Communications Between Services

Figure 2.1 shows various messaging protocols (N2, S1, S6, N35 etc..,) relying on Stream Control Transmission Protocol (SCTP) that are used to communicate between the micro-services in the function driven architecture employed by modern cellular network cores. Users communicating with cellular networks implicitly trust that the communications between these components which could be located within the same datacenter, or in a hybrid-cloud setting are secured. These communications include both the *control-plane* traffic and the *user-plane* traffic.

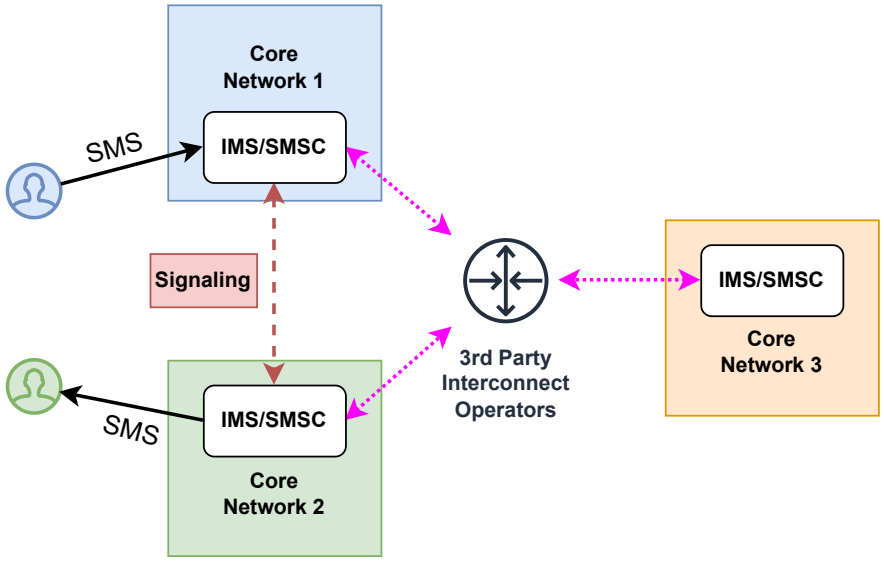


Figure 3.1: Direct signaling and hidden third party interconnect operators in cellular core networks.

SMS Center (SMSC) was introduced to the core network architectures in 2G and 3G allowing network operators to route messages between carrier networks supported by SS7 signaling. This evolved into a backward compatible IP Multimedia core network Subsystems (IMS) in 4G – establishing a standardized cellular core network service that allows for IP based multimedia services such as Voice over IP (VoIP), and text and multimedia messaging services. As a result of this architecture, users implicitly trust their cellular network provider with their communications

between subscribers. Cellular subscriber databases maintained by the SMSC or IMS services contain clear text record of communications from the sender to the receiver. As shown in Figure 3.1, signaling communications shown by the dashed *red* arrows between the core network 1 and core network 2 are necessary to deliver a SMS message between subscribers of two different cellular networks. The message details are available in plain-text to both cellular operators affecting user privacy for what users perceive as private messages between subscribers.

Signaling protocols such as Signaling System 7 (SS7), and DIAMETER form the core part of the control plane in a telecommunication network and is separated of the user plane. The older and still widely used SS7 signaling protocol supporting 2G and 3G networks allows trusted hand-off of communications from one carrier to another after necessary authentication and authorization. Communications between subscribers of different cellular networks are signaled between the network operator core networks allowing both parties, and intermediate third parties operating relays the ability to terminate the connections and learn the contents of the messages being signaled in plain-text. Compromise or breach of these infrastructure allows complete control over the control plane of the cellular network – enabling SMS redirection and account takeovers.

Despite the many innovations in cellular protocols over the last decade improving performance, security, and privacy, the ubiquitous, well supported, and widely used Short Messaging Service (SMS) standard continues to be insecure and sends messages through the network in clear text. Over a trillion messages which include sensitive content such as one time passwords (OTPs) used for authentications, password reset messages, banking transactions, and medical appointment reminders are sent every year from businesses to their customers. While the provisioning of phone numbers by the telecom operators through explicit hardware SIM cards are compelling reasons for businesses worldwide to rely on SMS for authentication and other communication, the widely used SMS standard allows telecom operators and various intermediaries between networks to view the message contents, compromising the sender and receiver's security and privacy.

While I make a case for SMS communications due to its ubiquitous and widely available support, the same security and privacy challenges exist for voice based communications and Internet connectivity services provided by the packet gateways operated by the cellular network providers. With the emergence of TLS in the web, intercepting the packet gateways to identify user packet flows would result in encrypted data flows. Through mechanisms such as deep packet inspection, or forcing UEs to use network operated DNS resolvers, operators can enforce various content filtering mechanisms preventing user access to the Internet. Cellular subscribers in most regions of the world implicitly trust their network operators to not intercept, modify or drop any messages and communications.

3.1.3 Hidden Entities in the Cellular Network Ecosystem

To reduce the complexity associated with pair-wise establishment of signaling between n core networks ($O(n^2)$), cellular network cores are connected by third party interconnect operators mimicing a *star* topology in a communication network as shown by the magenta dotted arrows in Figure 3.1. These signaling operators increase the number of networking hops required for a message to move from a source network to a destination network but simplify the operational complexity of cellular core networks which offload the routing complexity to the third parties. However, while these interconnect operators are invisible to subscribers they have complete visibility into clear-text communications between the intended subscribers across network cores. Compromise of these hidden networks leak user privacy. These risks are not theoretical and have been actively exploited by attackers silently for multiple years before being discovered in the SS7 hack affecting United State's largest 3rd party interconnect – Syniverse [71, 184].

In addition to these interconnects, operators providing API and authentication services for Internet applications or businesses are processed by hidden telemarketing or SMS gateways which connect both to the public Internet network and to telecommunication network backbones through

their private infrastructure. These popular web services are used by thousands of applications such as ride sharing, food delivery platforms and by end-to-end encrypted over the top messaging platforms like Signal messenger, or the extremely popular application – WhatsApp. Additional hidden network functions such as spam and abuse detection mechanisms, deep packet inspection, call and data records auditability, advertising networks, and integrations with services required by law enforcement, are either implemented directly by the cellular operator or are provided to them by third party service providers. This worsens the privacy gap since the contents of the messages are fully revealed to machine learning based spam and abuse detection models, or network traffic is used to identify and target subscribers with advertisements.

The telecom ecosystem and their services provided are complex, use insecure communication protocols, or are only encrypted from point-to-point allowing intermediate points in the network architecture full visibility into the contents of the communications. These protocols have previously been attacked by adversaries, and the network contain many critical players who are invisible to subscribers but yet route and process subscriber's data posing high security and privacy risks.

3.2 Implicit Trust in Hidden Internet Infrastructure

3.2.1 Trusted Default Configurations from Network Operators

Internet service providers and cellular network operators provide users access and connectivity services to the Internet. On successful authentication, an IP address is assigned (typically dynamically) by the service provider to the UE requesting access to the network. Dynamic Host Configuration Protocol (DHCP) servers are maintained by service providers to provide network configuration parameters such as the information about the default gateway, the network subnet mask, and the default DNS server configuration. A typical Internet user is typically oblivious to the existence of invisible infrastructure such as the DHCP servers or DNS resolvers. Due to the importance of DNS resolvers, large carrier networks and ISPs deploy privately managed DNS

infrastructure at their network perimeter. A historical omission of a discovery mechanism from the DNS ecosystem makes wider secure deployments of DNS services a challenge restricting the ability for automatic configurability to within the local network through the use of DHCP.

As mentioned in §2.2, DNS resolvers and infrastructure is critical infrastructure for a human-usable Internet. UDP based DNS queries are the most widely used mechanism for interacting with DNS resolvers and relay communications between the user and the resolver in plain-text, allowing the DNS infrastructure to have complete visibility into each user's Internet browsing patterns. ISP networks across the world enforce content filtering mechanisms either due to legal court mandates, regulatory policies such as Children's Online Privacy Protection Act (COPPA '98), or government enforced censorship policies due to religious, social conflict, or political reasons. Prima facie, the DNS infrastructure becomes a critical point for network interception and is supplemented by deep packet inspection, or IP address based filtering mechanisms. While the usage of private DNS resolvers which do not implement such filters is possible, users typically do not manually configure their DNS settings. Many private networks such as coffee shops, airport networks, and in-flight Internet networks do not allow users to configure custom DNS resolvers and prevent access to the Internet. As a result, users of these networks trust the default or *explicitly configured* DNS service providers to (1) provide valid responses to their DNS queries enabling them to browse the Internet, (2) safeguard their browsing activity from attackers, and (3) perform regular security audits and enforce strong privacy policy protecting their browsing activity.

3.2.2 Trusting the Trusted TLS Connections

The usage of Transport Layer Security (TLS) standardized the security for clients during communications with web, mail and other server infrastructure providing Internet enabled services. The ability for TLS to provide *authenticity*, *integrity*, and *confidentiality* prevented passive network adversaries from monitoring network communications between the user and the end host,

and active network adversaries from covertly modifying or tampering the network packets. As mentioned in §2.2, and §2.4, the successful adoption of TLS relied on large organizations such as Google providing browser software to mandate its usage, and penalize search rankings and reputation for those hostnames which do not enable support for secure communications between the client and the host.

Additionally, the development of open source and non-profit organizations such as Let's Encrypt, and CDN networks providing free TLS certificates improved the adoption of TLS on the Internet today [8, 92, 128]. The increased adoption of TLS, and the usage of CT logs resulted in clients implicitly trusting a set of certificate authorities forming the trust anchors. This collection of certificates is maintained differently for each operating system, and could be shipped independently by various application providers such as browser vendors, or core networking client stubs installed on the user devices. Critically, the guarantee of secure communications between a client and a server does not necessarily guarantee privacy. During communications with the server terminating the TLS connection, clients leak their IP address – which can be used for geolocation returning the information about the ISP, country, and sometimes the city information with reasonable accuracy.

TLS introduced *cipher suites* which are a collection of algorithms that can be used to secure the connection. Each entry in the cipher suite contains a tuple of the key exchange algorithm, the encryption algorithm, and the message authentication code algorithms. Since the introduction of TLS, the protocol has evolved with updates in 2006 adding various protections against cipher block chaining attacks and was later refined in 2008 resulting in TLS 1.2 upgrading the usage of insecure hash functions such as MD5 and SHA1 to SHA256 and expanding support for authenticated encryption ciphers such as AES GCM [282]. Recent updates resulting in the development of TLS 1.3 introduced mechanisms separating the key agreement and authentication algorithms and completely removing support for insecure cryptographic hash functions. The usage of older TLS protocols is formally deprecated by most modern browsers, or result into warning messages being

shown to the clients. However, older protocols are still prominently used to secure mail server communications and by low power Internet of Things (IoT) devices.

TLS handshakes contain a plain-text *ClientHello* message that is sent by the client to the server. The message contains the capabilities supported by the client and indicates the hostname as the server name requested. This message leaks user privacy and reveals to the ISP and any other on-path network adversaries about the intent of the connection. Deep packet inspection tools rely on this signaling to enforce content filtering or intercept and reset client communications [276, 293]. The Encrypted Client Hello (ECH) extension attempts to address this privacy concern by encrypting the initial *ClientHello* message such that only the intended webserver can decrypt the intended message. It does this by relying on the underlying DNS infrastructure to secure fetch the public keys needed for encryption. As a result, clients making these requests for TYPE65 records continue to trust DNS resolvers to resolve, and validate the communications during the traversal of the DNS hierarchy and to not modify the contents of the response.

3.2.3 Implications of Routing Infrastructure 'Built on Trust' on TLS Communications

The CAs that users rely on to keep their Internet communications authentic and secure, in-turn rely on hidden Internet infrastructure such as routers to successfully complete various challenges from the host requesting a certificate. Routing infrastructure on the Internet relies on specialized hardware known as routers, running the Border Gateway Protocol (BGP), and allows IP packets to be moved from a source towards a destination IP address through a collection of intermediate networks. Routers read packet header information and determine the next router to which the packet can be forwarded. The determination of *which* IP addresses to route to a network is defined by the *advertisements* made by different networks which are also referred to as Autonomous Systems (AS). ASes on the Internet network are dynamic and the BGP routing information sent via advertisements by a network to its *peers* are heavily trusted. This *implicit trust* in the routing

infrastructure allows adversaries to manipulate the routes IP packets take along the network either in attempts to passively observe the flows, or actively prevent traffic from reaching the destination.

BGP route manipulation, and route hijacking resulting into denial of service (DoS) attacks could happen either due to intentional attacks by adversaries or compromised networks, or unintentional and inadvertent mis-configurations. The usage of BGP Origin and Path Validation (BGPSec) allows ASes to cryptographically sign the BGP path update messages propogated between routers allowing a set of routers forming a path to validate the correctness of the path [209]. Resource Public Key Infrastructure (RPKI) allows ASes to cryptographically sign their advertisements as Route Origin Authorization (ROAs) objects allowing routers to only accept valid route advertisements to update their routing tables once the BGP path has been validated [66]. While the usage of BGPSec and RPKI could keep the Internet routing secure and trustworthy, compromise of the routing registry accounts (ARIN, RIPE etc..,) maintained by ASes can result in invalid ROA configurations being published by the AS resulting in IP addresses no longer being reachable and affecting network reachability and connectivity for users relying on the network. Recently, credential compromise to create and publish ROAs by a Spanish carrier network Orange, resulted in the network failures effectively reducing their Internet traffic by half [215].

Trust in routing is crucial for the issuance of certificate by CAs which are in turn heavily trusted to enable secure TLS communications on the Internet. Hijacked communication between CA and the hostname could result in attackers successfully completing various domain validation and HTTP based challenges posed by CAs and obtain TLS certificates on the host's behalf [55]. Despite the possibility of such attacks being known since 2018, real world BGP hijacks in 2022 enabled the issuance of certificates by ZeroSSL resulting in the KlaySwap cryptocurrency attack [289]. Additionally, the prevalence of government CAs in the root trusted certificates forming the trust anchors increases the risk for national intelligence efforts and countries with heavy censorship to gain control of issued certificates and then perform MITM attacks on users. The use of multi-

vantage point domain validation infrastructure has been shown to thwart a majority of these BGP attacks and is actively used by Let's Encrypt [56].

3.3 Implicit Trust in Today's End User Privacy Applications

Various widely used and popular Internet enabled applications provide users security and privacy while interacting with resources on the Internet. These range from anti-virus and Internet security software installed on user devices, Virtual Private Networks (VPN) applications allowing users to protect their privacy, and E2EE messaging and communication applications. While all these applications keep users safe and private on the Internet, they require the user to implicitly trust them to do so which is not ideal.

Typical Internet security and antivirus software installed on user devices require users to install custom root certificates of the antivirus software vendor, which are typically regularly audited. Enabling users to customize their trust anchors and modify their list of root trusted certificates is considered to be insecure due to the possibility of nation scale social-engineered attacks that could be used to compromise user security and privacy by intercepting all their device communications. Users installing these software typically do not understand the implications of doing so and this unfamiliarity has been previously exploited to launch the SMS campaign asking citizens of Kazakhstan to install a *national security certificate* on their devices [276]. Despite being able to orchestrate similar attacks in case of breaches, antivirus software are heavily *trusted* by the majority of those who install it and at-times are installed at the organizational level by IT administrators to secure enterprise networks and their employees while also allowing them to observe and monitor network traffic to protect the organization.

VPN applications are used and configured by privacy aware users to protect their network traffic from their country ISP – typically to circumvent content filtering or censorship, or hide network traffic from other passive on-lookers on the network path. As a result, all network traffic

is sent over an encrypted tunnel between the client and the VPN service provider who relays the communications on the client's behalf. As a result, the intermediate VPN services shift the trust from relying on DNS and the served TLS certificates to trusting and relying on the VPN. The usage of VPNs however does not necessarily prevent servers from identifying the users who could continue to use other indicators and priors to establish the mapping. The privacy benefits are improved as more users rely on the same VPN infrastructure and route their traffic through them effectively gaining privacy through the introduction of noise. However, despite the increasing adoption and widespread usage of VPNs, recent research efforts indicate that popular VPN protocols are vulnerable to fingerprinting [353]. Active government surveillance and attempts to crackdown on VPNs by ordering providers to handover user data and forcing user data to be stored for 5 years or longer raise significant concerns around user's trust of their Internet service provider and the various services they interact with using their configured VPN network [154].

Another widely used application focused on providing security and privacy are end-to-end encrypted communication platforms such as Signal, or WhatsApp among others. Advances in cryptographic protocols resulted in E2EE communication platforms deploying double ratcheting algorithms [266], and using an Extended Triple Diffie-Hellman handshake for key agreement (X3DH) [219] allowing two parties to establish a shared secret key while mutually authenticating each other. To guarantee correct key agreements, the two parties need to physically meet and validate their shared secrets out of band. Such explicit efforts at validation are rare, and many users trust the public keys sent by the server infrastructure of the communication platform maintaining the identity of the respective users. As a result, users face risk of equivocation attacks from centralized providers of such infrastructure – who can present diverging views resulting in a MITM attack orchestrated by the provider, possibly on behalf of a law enforcement request. While popular platforms deny such actions, no technical guarantees exist to prove or disprove them – recently enabled in WhatsApp through deployment of Auditable Key Directories [216, 226].

E2EE and Internet based applications often rely on a source of digital identity – often provided by cellular operators due to the unique nature of hardware SIM based identifiers. E2EE messaging applications like WhatsApp, Signal, use phone number based authentication as a part of the *bootstrap* process used to create a user account prior to generating a cryptographic key pair in the hardware trusted execution enclave of the user device. As a part of the authentication process, a single time short lived one time password (OTP) is sent from the servers via a third party gateway with both Internet and cellular backbone connectivity, which is routed to the user's network operator who is responsible for finally delivering the message that is used to complete the authentication process, eventually establishing an identity on the messaging platform. Such authentication workflows are used to prevent sybil attacks where millions of fake accounts would be created otherwise by attackers. An often overlooked portion of security and trust models is this interconnect and the reliance of Internet based end user applications on insecure and highly centralized nation scale cellular network infrastructure. The operators of government controlled cellular networks can as a result, read the sensitive OTP messages being sent by the E2EE messaging provider to the intended subscriber. These messages could be dropped by the operator preventing users from using E2EE applications they're attempting to gain access to [68], or could be intercepted resulting in account takeovers.

Summary: Without many user's knowledge, sensitive information which is typically protected in TLS enabled communications on the Internet are relayed in plain-text and become interceptable while using carrier networks. Due to the nature of mobile devices providing both Internet and cellular service connectivity to users, it becomes difficult for a user to distinguish the nature of data flows within the network. The heavy reliability of Internet applications and services on cellular infrastructure which does not provide the same levels of security and privacy due to its centralized nature challenges the nature of trust in networking as the boundary between the two blur and infrastructure becomes increasingly hidden.

Chapter 4

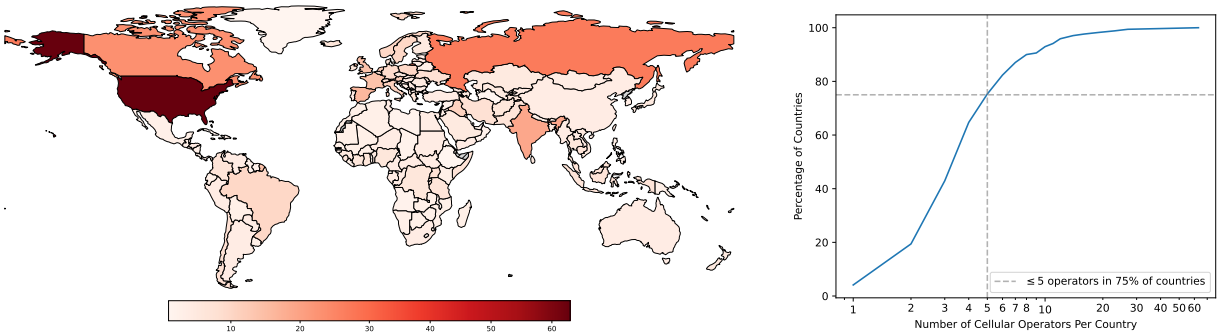
Addressing the Long Tail of Internet Access

Organizations are increasingly turning to unlicensed spectrum and host neutral architectures to provide network services through private 5G/LTE or Community Cellular Networks to address the challenges of connectivity in both rural and urban Internet affordability and connectivity. To the subscribers of these services, their widespread deployment and active usage comes with two key challenges – affecting their (1) security, and (2) usability.

4.1 The Case for Decentralizing Cellular Networks

The evolution of cellular network core infrastructure has resulted in the industry adopting service based design. While this enables operators to identify bottlenecks and scale their services to provide efficient and quick access to the Internet and other cellular services to their subscribers, the design becomes increasingly complex and difficult to maintain for smaller scale individual network operators. As the usage of the cloud increases and private 5G and LTE networks continue to increasingly be deployed, it becomes critical to enable users to move between networks securely and without interruptions to their quality of service. As mentioned in §3.1, nation scale cellular networks today are heavily trusted and any failures – intentional or unintentional result in severe adverse impacts to its users and the economies that rely on these connectivity services.

While large nation scale carrier networks have enabled economies of scale, they've raised concerns about security and privacy due to the amount they're heavily trusted. Additionally, the users of such networks are heavily restricted due to the lack of market-competitive alternatives in the most cases providing them a market to *choose* from a large set of service providers. We analyze the data of active network operators globally based on active Mobile Country Code (MCC) and Mobile Network Codes (MNC) assigned to operators by the International Telecommunication Union Telecom standardization sector (ITU-T). As shown by in Figure 4.1, we observe that subscribers in 128 countries (75% globally) have less than 5 cellular operators to choose from indicating the monopolistic or oligopolistic nature of network infrastructure providers globally. Such co-location of the network infrastructure makes nations vulnerable to technical failure of infrastructure and malicious attacks by adversaries. Additionally, it opens up tremendous possibilities for abuse of power by nations over its citizens who have *no other choice*.



(a) Number of cellular network operators per Country

(b) CDF of cell networks by country

Figure 4.1: Distribution of number of cellular network operators globally.

As a result, it is important to establish alternative and reliable communication mechanisms and decentralize the centralized nature of this infrastructure. The deployment of thousands of small scale private or community cellular networks is no longer a pipedream due to the relatively low cost of communication hardware, production grade open source cellular core network software, and

the democratization of spectrum through the Citizens Broadband Radio Service (CBRS) allowing hyperlocal cellular network deployments in both urban and rural areas supporting a range of use cases from enterprise settings to enabling affordable Internet access. Prior research efforts have argued that the deployment of such networks is cost-effective, especially in regions where it might be prohibitive for large network carriers to deploy their infrastructure [39, 145, 147, 182], enables a local community for repair and maintenance [170, 171], and empowers the community by giving more control over the use of network resources [175].

While decentralization allows increased resilience due to the availability of alternatives and reduces individual control of the large network providers enabling community ownership, it also raises significant challenges for security, and how trust is established in a network. The use of blockchains due to their tamperproof nature, cryptographic guarantees, and need for consensus among participants have given rise to interesting and newer trust models which are worth exploring in a cellular network setting.

4.2 Coordination and Interoperability Challenges

In today's nation scale cellular networks, the interoperability between networks is enabled through pre-established agreements and is commonly referred to as *roaming* [214]. These roaming agreements between the remote mobile network and the subscriber's home network is established either directly between the two parties or through partnerships with a network aggregator. While these roaming agreements might be easy to establish between a small set of carriers, the same approach would be a non-starter for a network setup involving hundreds if not thousands of private smaller core networks. Additionally, roaming configurations require network operators to be always available to establish secure connectivity between network cores which may not always be possible in disconnected and individual single cell rural deployments of these networks.

Roaming in the traditional national scale cellular context is established by a serving roaming

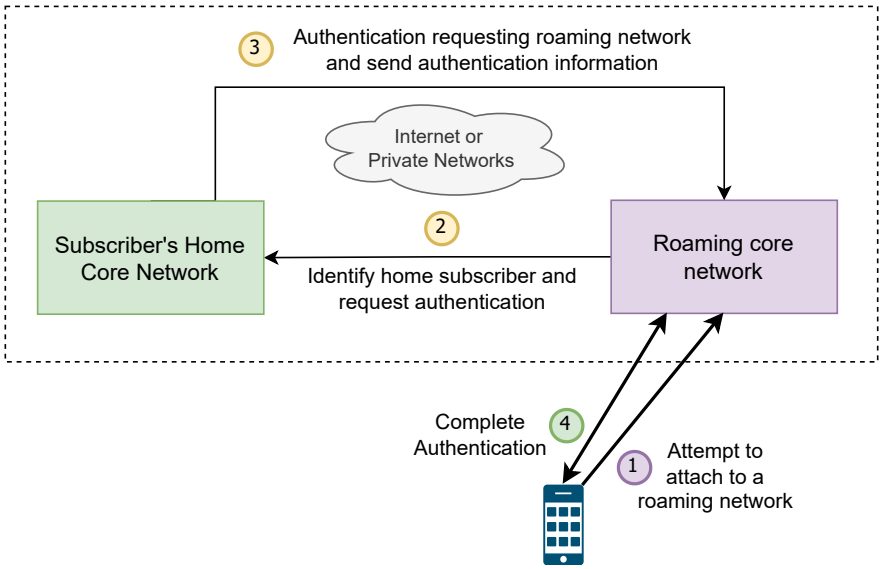


Figure 4.2: Overview of roaming in traditional cellular network between two core networks

network contacting the home network of the subscriber for authentication information to challenge the user device's request to attach to the roaming network. This enables the user device to perform authentication with the home network's core identity services while being routed by the roaming network provider as shown in Figure 4.2. The subscriber's home core network then completes a bi-directional authentication and informs the serving network about the status of the authentication with a session token to allow the UE to connect and obtain services provided by the roaming network. Due to the nature of the pre-established roaming agreement establishing business trust relationship, the inter-provider roaming protocol indicates the authenticity of the user with the roaming network. Additionally, these messages might be relayed through third party intermediaries who interconnect different network operators where direct peering links are not established. Once the core networks authenticate each other, and home subscriber network assists roaming network to authenticate the user, the network operator sends the device configuration information provisioning access to an Internet packet gateway allowing Internet access.

While the current roaming practices might work for large nation scale cellular operators,

or mobile virtual network operators (MVNOs), it becomes particularly challenging to establish communications back to the home network from a roaming network core when hundreds of smaller such operators exist or in the case of rural networks where users might want to access local in-network services when individual network cores might be disconnected from the backhaul Internet services. This is both due to the challenges in providing high speed inter-operator network connectivity and the difficulty in establishing roaming agreements with each available network operator.

Due to various billing requirements – typically based on the usage of bandwidth by the UE requesting the roaming services, the roaming network cores are typically configured to tunnel user traffic to the home network core where the PCRF function in the network manages billing. The results from the individual PCRF functions of both networks are compared and audited generating financial reports which are periodically settled by the operators involved. The standards for roaming are under-specified, and is operator dependent due to its high complexity. To address these challenges, we propose the establishment of a consortium blockchain of smaller scale network operators. The overall design also allows nation scale network operators to inter-operate with individual network cores.

4.3 Reducing Trust Requirements Between Carrier Networks

Envisioning a world with thousands of smaller cellular network operators, a key challenge would be to establish communication between the core networks without explicit business agreements between the network operators. Blockchain networks address the challenges of enabling disparate nodes to coordinate and show great promise at addressing the problems associated with coordination and interoperability. In a decentralized network relying on blockchains, each network core typically is configured to operate as a *peer* managing their own organizational identity. Message flows between core networks are wrapped as cryptographically signed *transactions* which are

grouped together into a *block* created by the participating nodes of the network. These blocks are cryptographically linked to the previous block providing guarantees of transaction ordering and authenticity, by establishing a tamper resistant append only logs of transactions agreed upon by mutually untrusted entities.

Algorithm 1 Node Setup

$\Pi_R \leftarrow$ External Proof
$N_i \leftarrow$ Node Unique ID
$(\text{SK}, \text{PK}) \leftarrow \text{KeyGen}()$
$\sigma_r \leftarrow \text{Sign}(N_i \| \Pi_R, \text{SK})$
$\textsc{Register}(\text{PK}, N_i \| \Pi_R, \sigma_r))$

Algorithm 2 Tx Issuance

Require: $T \leftarrow \{\}$
    $T[i] \leftarrow \text{PK}_i \; \forall \; i \rightarrow \text{PK}_i$
$m \leftarrow$ Encoded Request
$\text{Tx} \leftarrow \textsc{Enc}(m, T[i])$
$\sigma_{\text{Tx}} \leftarrow \textsc{Sign}(N_i \| \text{Tx}, \text{SK})$
$\textsc{SubmitTx}(N_i \| \text{Tx}, \sigma_{\text{Tx}})$

Algorithm 3 Tx Validation

$i, \text{Tx}, \sigma_{\text{Tx}} \leftarrow \text{RecvTx}()$
if $i \notin T$ **then**
    return (false, ϕ)
end if
$\text{PK}_i \leftarrow T[i]$
$E_m \leftarrow i \| \text{Tx}$
$b \leftarrow \text{Verify}(E_m, \sigma_{\text{Tx}}, \text{PK}_i)$
if b **then**
    if $i \neq N_i$ **then**
        return (true, ϕ)
    end if
    $m \leftarrow \textsc{Dec}(E_m, \text{SK})$
    return (true, m)
end if
return false

The usage of blockchains in the setting of cellular network cores as a coordination layer allows auditability of services provided which can be used for billing and reduces the need for independent audits being carried out. However, public blockchain (Bitcoin, Ethereum Classic, etc..,) settings require majority of the participants to be honest, but come with the trade-off of

increased computational overheads due to the *proof of work* nature of consensus. In a heavily government regulated setting such as cellular networks, we argue that such extreme notion of distrust is unnecessary and could be relaxed – allowing the deployment of faster, practical, and less compute intensive *permissioned* blockchains which pose a relatively small barrier of entry for new operators joining the network. In practice, the proof of registration of the operator's business could be used to enable their respective cores access to the permissioned blockchain network. The pseudocode for a new node registration is mentioned in Algorithm 1.

Cellular operators participating in this permissioned blockchain network could coordinate between each other by issuing a transaction on the blockchain to be included into a block. Algorithm 2 presents a generic interface exposed as an API from the node to the core network functions in the EPC or 5GC. In the case of roaming, this transaction effectively can be an encoded request for authentication information necessary to validate a particular user. The issuance of such a transaction to be addressed by a peer instance i is proof that a user device attempted to connect to the roaming network and requested roaming services through the provider. The usage of the blockchain indicates that core networks which are not physically peered with each other still obtain the message through an intermediate node participating in the network. The knowledge of public keys associated with the participating nodes of the network allows the transaction contents to be encrypted for receipt only by the intended recipient core network and allows participating nodes to achieve consensus over encrypted encoded requests as transactions. Similarly, a response to the transaction is linked to the hashed identifier of the originating transaction allowing auditability of the records. All receiving nodes in the consortium receiving the transaction, validate the authenticity of the transaction by verifying the integrity of the signature over the encrypted message contents by the issuing node as shown in Algorithm 3. Once validated, if the transaction contents are intended for the peer verifying the transaction, it is decrypted and the encoded requests are processed accordingly by the business logic implemented with the cellular cores.

In the context of roaming, the usage of consortium blockchains removes the explicit need for establishing business roaming agreements and performing regular audits of the billing and security infrastructure. Due to the nature of blockchain driven consensus and data sharing, it allows cellular operators to share information with each other despite not being immediately connected to each other, allowing existing Internet gateways and cloud services to be used. The democratization of cellular network deployments allow individual users to quick move between service providers offering greater choice and freedom.

4.4 (Un)Intended Consequences & Performance Implications

The usage of cryptographically chained append only datastructures in blockchain networks require each node participating in the network to maintain a copy of the same information resulting in increased storage costs for small network operators. Additionally the mechanism of using blockchains transactions to communicate messages that would otherwise have been achieved over a connection oriented protocol between the participating network cores result in significant latency overheads because of the time taken for achieving consensus. Such latency overheads are typically in the few milliseconds in traditional roaming networks and switching to a blockchain based decentralized architecture would adversely impact subscriber's experiences.

An intended consequence of a decentralized architecture is to enable the usage of public key cryptography in cellular infrastructure which has traditionally only used symmetric key cryptography based techniques for authentication and protecting the integrity and confidentiality of the messages being sent. The usage of public key cryptographic techniques brings together the Web based ecosystem and the cellular ecosystem further blurring the differences between the two. The cellular ecosystem as a result could leverage existing public key infrastructure and seamlessly interoperate and benefit from existing transparency infrastructure while offering greater visibility and insights into cellular networks which did not exist previously.

The usage of public keys enables the introduction of discovery mechanisms allows roaming networks to discover the home network of the subscribers requesting the service. An intended consequence of this design is the ability to selectively encrypt messages that can only be read by the intended recipient while allowing other peers to reason about the integrity and correctness of the encrypted information during consensus. This is possible due to the high level nature of transactions which are *wrappers* over any mutually agreed upon message encoding schemes supporting custom business logic between two parties and could also be used for exchanging secure standardized message formats. This design improves privacy for subscribers by preventing any intermediaries from the intent of the transaction and the contents of the messages. It also allows for the eventual usage of threshold cryptosystems where disputed messages are efficiently decrypted by a honest threshold during audits improving trust and privacy [32, 49].

Decentralizing the cellular core network also allows for greater innovation in network infrastructure and democratizes the cellular ecosystem allowing for additional service providers and existing cellular ecosystems to thrive while making them more transparent and visible. Real world deployment of consortium blockchains in cellular networks in India – focused on addressing spam in promotional messaging, enable auditability and optimize business processes improving user experiences and are shown to be effective and operational at scale of billions of users [312].

An (un)intended consequence and the push against decentralization of critical infrastructure are the increased efforts associated with lawful intercept and national intelligence operations. While such efforts might be considered unconstitutional by many due to privacy invasion, they're considered as a critical necessity by many others. Keeping in mind a positionality bias – while alternative views against decentralization make fair points and are challenges which need to be addressed by future interdisciplinary research efforts, I strongly believe that the improvements to infrastructure resilience, user privacy and security, and the ability to circumvent Internet censorship in many regions of the world with such architectures far outweigh its limitations.

4.5 Implementing Decentralized Cellular Authentication

A key challenge with enabling core network services such as authentication is the latency overhead associated with the parties communicating through blockchain transactions. To overcome the challenge, we investigate the standardized authentication mechanism used in LTE and 5G core networks enabling user devices to connect to the core networks [17, 48, 191]. User sim cards contain symmetric keys K and introduce themselves to the radio base stations using their IMSI (4G LTE), or a concealed identifier (SUCI) resulting in an authentication request being sent by the MME to the HSS database in 4G, or the AMF through the AUSF to the UDM as shown in Figure 2.1. The HSS or the UDM return an authentication response vector containing the authentication information AUTH, an expected response from the user device XRES or its hashed variant in 5G (HXRES), and an intermediate key K_{ausf} which is used to derive other intermediate keys used for securing user device communications with the radio base stations and core networks. On closer inspection, we identify that these tuples (AUTH, XRES, K_{ausf}, RAND) and the associated randomness are the cryptographic bits required for successful authentication.

A key insight we derive is that due to the replay attack resistance provided by one-time use sequence numbers (SQN) in the SIM card, it allows home network operators with the symmetric key of the subscriber to *pre-compute* the authentication vector tuples and encode them into transactions on the consortium blockchain network. By encrypting specific vectors with all parties in the network, it allows the home network operator to issue a fixed number of transactions per subscriber anticipating user roaming. As a result, the roaming network operator who has the authentication vectors can challenge the user device to prove its authenticity and match the expected response (or hashed expected response). This also removes the potential bottlenecks and performance implications that transaction based message passing but incurs additional storage overheads for the core network. The usage of the tokens issued in transactions are reported when used creating an auditable trace that can also be used for billing.

To address these challenges we propose *dAuth* – a decentralized authentication mechanism that allows users to seamlessly roam between different network cores preventing the need for the individual network cores to coordinate on demand, and maintaining security and privacy while allowing UEs to roam in different networks without sharing sensitive symmetric key material. The proposal also addresses the challenges of reduced quality of service when roaming since all user plane traffic is routed back to the home network and allows for local breakout, providing users the advantage of existing optimizations such as load balancing and content delivery networks edge availability.

We implement a distributed HSS between multiple network EPCs where users may roam between these networks without prior roaming agreements or key sharing and provide the results from of our distributed-HSS-based authentication. The implementation for the distributed authentication builds on the open source LTE/5G EPC stack *Open5GS* [257]. *Open5GS* is an AGPL licensed, C based open source implementation of the 3GPP Evolved Packet Core (EPC) of an LTE network with support for SIM cards using the Milenage algorithm as detailed in the ETSI Technical Specification [1, 257]. Open5GS supports the LTE specification as provided in the 3GPP Release 14 [4]. The implementation currently does not support the standard roaming procedures used in LTE. The blockchain between the EPCs is implemented with an open source consortium blockchain *Hyperledger Sawtooth* with all the nodes running the PBFT/PoET consensus protocol [74, 163, 254].

Trust Model: As in standard cellular networks, the HSS contains a copy of the symmetric LTE Key K_i which is also in the SIM card purchased by the customer, thereby establishing a trusted relationship between each cellular provider and their subscribers. The regulatory protocols in place for registration of the carrier and legal requirements will act as a trust barrier, on passing which the carrier or Mobile Virtual Network Operator (MVNO) can join a Sawtooth consortium network. We may assume that transactions and networks can be rightfully moderated by regulators who could have a *read* capability in the network. In our model we assume that all cellular operators, which

are peers in the Sawtooth network, deploy the same Sawtooth *transaction processors* (discussed in more detail later) for performing the required validations. They establish secure connections which can be achieved with pre-shared network security keys to establish a secure communication channel. The peers issuing the *transactions* cryptographically sign their messages using elliptic curve cryptography which can be validated by other members in the network. The Proof of Elapsed Time *(PoET)* consensus protocol is executed on a Trusted Platform Module (TPM) chip like the Intel Software Guard Extensions (SGX) enabling application developers to use trusted code and produce signed attestations preserving confidentiality and integrity of sensitive data.

PoET: is a consensus algorithm developed by Intel for the Hyperledger Sawtooth blockchain which aims to reduce the wasteful work that occurs in traditional Proof of Work *(PoW)* systems like Bitcoin/Ethereum [163, 245, 352]. PoET creates a lottery similar to PoW systems but instead of solving a cryptographic puzzle to create and propose a block it uses TPM hardware to sample for a wait time from an exponential distribution sampled from trusted code running inside the SGX enclave. The winner waits for the specified time and proposes the block to the network if no block is found, or watches the network and helps resolve potential forks favoring the chain with a lower aggregate waiting time. The result of the wait in the form of the Trusted Execution Environment (TEE) attestation is validated by the other participants of the network. SGX prevents user-defined applications from accessing secure areas of the processor, thereby preventing attacks from spoofing or modifying the attestation received from the TEE device. In case of a detected attack, PoET implements a statistical z-test on the block publication rate of the node forcing it to not win the lottery, thereby preventing it from publishing too many blocks in the network.

Every sawtooth node consists of multiple processes packaged as services. The main sawtooth service is the *validator* which is responsible for validating transactions or batches of transactions, combining them into corresponding blocks and coordinating communication between various other validators on different sawtooth nodes in the network. *Transaction families* are application-

specific business logic separating the various types of validations that need to be performed by the validator. The *Transaction Processor* validates the transactions based on the transaction families the transactions belong to. Multiple transaction processors connect to the validator and report the status of the transaction belonging to a specific family. A typical sawtooth installation involves multiple default transaction processors like the identity processor and the settings processor which are run as services to aid the validator. The global state of all the transaction families are presented in a single instance of a Merkle-Radix tree which is present on each validator. Each transaction family and the corresponding transaction processor can write or read from this global state using a Radix Address which is computed as SHA512(Family Name)$[0:6]$||SHA512(Key)$[-64:]$, taking the first three bytes of the family as the namespace and creating a unique identifier for a *key* defined according to the application logic of the transaction family.

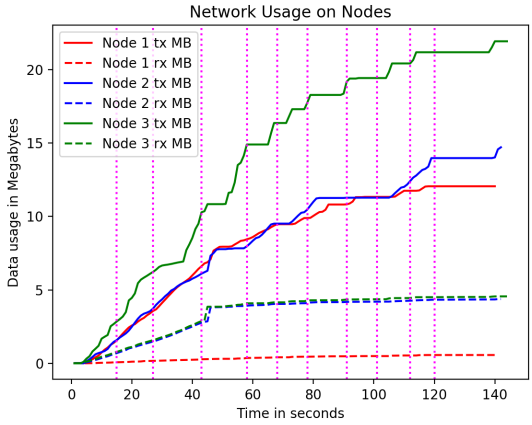
We implement our custom transaction issuing client as *ccellular*, and a corresponding transaction processor as *ccellular-tp*, as a *systemd* service which is registered to the validator and deployed. Similarly, the *ccellular* client is deployed on all the nodes in the network. *ccellular* is available as both a command line client and a module written in *python*, and is used to issue transactions or read the data corresponding to a given mobile identity (IMSI) in the sawtooth global state. *ccellular* is responsible for correctly encoding transactions into the format that the sawtooth validators understand in addition to performing the corresponding read operations. *ccellular-tp* is a transaction processor corresponding to the *ccellular* transaction family, and is responsible for receiving the transactions and performing state changes on the blockchain global state. The *ccellular-tp* implementation consists of database listeners similar to triggers listening to the changes in collections in Open5GS MongoDB data store. Updates to the MongoDB data store from Open5GS are converted into transactions by invoking the *ccellular* client and published to the network which are validated by the transaction processor updating the corresponding blockchain global state and the MongoDB databases of other network cores on the network.

4.5.1 Evaluation

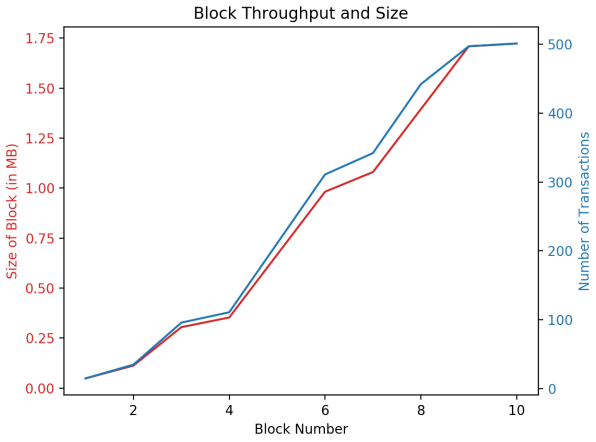
We evaluate the system along various metrics using a test network of three core networks, one with an attached eNB running on a fourth machine (but connected via the local network to one of the cores). Two of the three machines are Zotac BI325 Next Unit of Computing (NUC) mini-computers running Ubuntu 18.04 with 1.6 GHz Intel Celeron N3160 CPUs, one with 8 GB memory (Node 1) and the other with 4 GB (Node 2). The third is a Dell Optiplex 780 desktop computer also running Ubuntu 18.04 with a 2.83 GHz Intel Core 2 Q9550 CPU and 4 GB memory (Node 3). The fourth machine which runs only the eNB is a Dell Optiplex 7010 with a 3.40 GHz Intel Core i7-3770 CPU and 8 GB of memory, and the eNB radio is attached via USB 3.0 to the SuperSpeed (SS) ports. The eNB is implemented using the *srslte* open source eNB software [321], and runs on an Ettus USRP B205 software-defined radio (SDR); A Moto C Plus Android smartphone running Android 7.0 is used as the UE. The SIM cards, purchased from a manufacturer through the online store Alibaba, come with smart card functionality (similar to chip based credit cards) and run a Java Virtual Machine (JVM) with hardware guards and supports various standardized cryptographic algorithms used in LTE and 5G.

We evaluated the sawtooth network and the transaction processor by issuing 501 independent transactions as authentication vector updates in the database. These transactions were issued to the sawtooth network and validated before triggering a corresponding database write in the roaming HSS databases. In our experiment, sawtooth batched the transactions into 10 blocks of varying lengths as shown in Figure 4.3b. The default maximum batch size for transactions to be included in a block is 200 but we only see some blocks with maximum size of 100. The total experiment lasted 120 seconds from the initiation of the first transaction resulting in an average transaction throughput of approximately 4-5 transactions per second.

The total network bandwidth between Nodes 1 and 2, measured via iperf, was 118 Mbps, limited by the WiFi backhaul of Node 1; likewise, the bandwidth between Nodes 1 and 3 was



(a) Sawtooth Throughput: Plot of Megabytes (MB) sent (tx) and received (rx) by each of the three Sawtooth nodes as a result of 501 transactions sent by Node 1 at the same time. All transactions were committed in 120 seconds. The times of block commits are shown as the pink lines. The average throughput over all transactions was around 0.015 MB per second, or 0.12 Mbps.



(b) Sawtooth Throughput: Plot of block size and number of transactions committed for the same 501 transactions as run in Figure 4.3a.

Figure 4.3: Evaluation of blockchain based decentralized LTE core authentication

measured at 117 Mbps. The bandwidth between Nodes 2 and 3 was measured at 934 Mbps, as both were connected to the same wired LAN. Sawtooth used 0.12 Mbps on average, a surprisingly high bandwidth, but small compared to the total available. In the future, we aim to maximize the network bandwidth availability for maximizing throughput of the system.

These experiments on extremely resource constrained hardware indicates the viability of decentralizing core network functions and their ability to operate in rural community cellular network regions with intermittent network connectivity.

Chapter 5

Understanding the Internet's Critical Long Tail

Today, most secure web communication takes place over HyperText Transfer Protocol Secure (https). Using Transport Layer Security (TLS) to encrypt http requests and responses, https provides users with message *authentication*, *integrity* and *confidentiality*. Many elements of https usage have been explored, with previous work focused on measuring the cost of https [246], analyzing the certificate ecosystem [118], and the examination of https adoption in the web of 2017 [128]. While the most recent measurements by Felt et al. [128] focused on adoption of https using the Alexa top 1 Million dataset, many critical web resources are unlikely to fall within this dataset, such as websites run by local and national governments. Such sites, often serving smaller geographic regions or countries without a large web presence, are trusted with holding sensitive user data for civic functions or providing information such as local infectious disease numbers.

Across the world, government websites are expected to be reliable sources of information, regardless of their view count. Interactions with these websites often contain sensitive information, such as identity, medical, or legal data, whose integrity must be protected for citizens to remain safe. Prior case studies have shown that citizens visit local county government websites for a wide range of services including job openings, local demographics, budgets, meeting minutes, details of contracts and their summaries, and for official contact information of their elected

representatives [28]. Research also shows that websites providing quality e-services help build trusted relationships between citizens and their governments; further, low-traffic local government websites such as utilities, water *etc.*, while not present in top million lists, are still actively used in citizens' daily lives [326]. Attackers therefore may target government sites to disrupt critical infrastructure, steal identifying data, disenfranchise citizens and influence politics, or decrease their trust in the government. Providing secure access to local ".gov" sites should be of high priority for governments.

5.1 Datasets & Tools

Datasets of websites exist on the Internet for research use, including the Alexa million datasets which rank sites by popularity, the Cisco million [81] which ranks by traffic volume, and the Majestic million [176]–an open source version of the Alexa million since its acquisition by Amazon. Tranco, another public list, attempts to provide a more stable ranking for web measurement avoiding the flux of prior datasets [205]. Le Pochat *et al.* note that only 49% of the domains in the Umbrella datasets are available, responding with a success status code of 200, as are only 89% of the Majestic million [205]. Our work uses these datasets as a seed set, which we then expand through web crawling, Amazon Mechanical Turk tasks, and hand-searching domains to increase the number of unique measurable government websites from 27532 to 135408. This is a substantial increase from government websites in existing datasets, and forms the basis for our analysis. In prior work, tools like ZMap and CFSSL have enabled researchers to perform large scale studies on Internet hosts [85, 116].

Our analysis is different in that prior efforts largely focus on the "head" of the Internet, *i.e.* popular domains as found in top million lists. In this work, we explicitly include the "long tail" of government websites as they are especially critical to user safety but do not commonly appear in the top million lists. Mirian *et al.* similarly measured https among general sites outside the top

millions, finding that services providing free certificates such as *Let's Encrypt* improve overall adoption of https and that general web domains also use *Let's Encrypt* four times more than other CA authorities [230]. We observe that Let's Encrypt is also the most popular CA used by government sites globally, though not in every country. Prior studies have tried to understand the root causes of https certificate errors in Chrome [10] and analyze trust models in CAs [13, 125]. Others focus on challenges in the certificate ecosystem, the need to make them more auditable, and ways that CAs could be incentivized using insurance models with benefits negotiated between CAs and domains [117, 118, 150, 222, 260].

5.2 Measurement Methodology

We begin by generating an initial "seed" list of government hostnames by merging the publicly available top-million datasets, including the Majestic Million dataset, Cisco top 1 Million dataset, one historical copy of the Alexa top 1 Million dataset published in August 2019, and the Censys research dataset produced by the University of Michigan and made available through Google BigQuery [116, 295]. This merged dataset of hostnames is then filtered and de-duplicated to include only government websites. We separate government and non-government sites through a regular expression filter for hostnames using standard government formats. A popular format used by many countries is .gov.country-code, and all countries except the United States use only one domain extension. However, the USA uses both .gov.us and .gov for official government purposes, in addition to a dedicated federal .fed/.fed.us and military .mil top level domain (TLD) without the "us" country code.

Government domain names and extensions depend heavily on countries' primary languages. Countries with French as a primary language often use *.gouv*, and those with Spanish use *.gob* followed by country code. Kenya, Indonesia, Japan, Korea, Thailand and Uganda use *.go* followed by the country code. Some countries use *.gub, .govern, .government*, and *.guv*, New Zealand uses

.govt and Switzerland uses *.admin*. We filter hostnames in the dataset using these known expectations and exceptions, along with country code extensions, as a conservative filter with high precision but limited recall. This was decided to ensure that our list was comprised of only government websites. For example, environment.gov.au, geoportal.capmas.gov.eg, stats.data.gouv.fr & www.pwebapps.ezv.admin.ch are valid hostnames because they follow the format of a valid government domain name extension followed by a country code, making them valid ccTLDs included in our scan. We expanded this initial list through three separate mechanisms: 1) crowdsourcing local hostnames using Amazon Mechanical Turk, 2) crawling the hostnames in our list, and 3) hand-curating a set of government hostnames.

5.2.1 Crowdsourcing Through Amazon Mechanical Turk (MTurk)

Seeding with sites from the top millions inherently biases our results towards larger or more connected countries. To combat (but not entirely remove) this bias, we used Amazon's Mechanical Turk (MTurk), a popular crowdwork platform [187], to publish tasks for finding government websites for countries where we had only a few or no hostnames. Each task asked a worker to enter up to six URLs from a specific country, with USD 0.60 paid per task. To encourage site diversity, we asked workers to find different categories of government sites. The categories were: the National Government (or the Presidency if no national government site was available), Public Health (or a government News/Media site if none available), Taxes (or Finance Ministry if none available), Immigration or Travel, and any 2 different departments not covered. The tasks were completely anonymous with no repeat responses allowed from the same worker. The only demographic information queried was a binary Yes/No indicating if the worker was from the country in the issued task.

We published tasks for countries with less than 11 hostnames in the seed list, including Andorra, Chad, Chile, Democratic Republic of the Congo (DRC), Costa Rica, El Salvador, Guatemala, Iceland,

New Zealand, Nicaragua, Panama, Tanzania, Thailand, Tonga, Greenland, Western Sahara, Falkland Islands, Puerto Rico, New Caledonia, Solomon Islands, Northern Cyprus, Somaliland, Kosovo, South Sudan, and Niger. We received 108 responses, of which we accepted 75 after manual inspection. 11 workers self-reported as being from one of these countries. They were: 4 from Greenland, 2 from the Democratic Republic of the Congo (DRC), and 1 each from Andorra, Costa Rica, New Zealand, New Caledonia, Solomon Islands and Kosovo.

We obtained a total of 199 unique hostnames from the 108 MTurk tasks we issued, with 61 already in the seed list. 138 new hostnames were added to our seed list, bringing the size to 27,794. The usage of MTurk based crowd sourcing with respondents in the countries enabled the discovery of these websites which were typically in the respective local languages and did not follow the government TLD suffixes which were typically expected in our discovery mechanism.

5.2.2 Crawling Government Websites Globally

We built a web crawler for the above seed list (inclusive of added MTurk hostnames) that visits every hostname, gathers all links on the page not yet seen by the crawler with a valid country code extension (according to ICANN [164]).

The crawler follows the links for 7 levels of depth before terminating the crawl for that hostname. The crawler began with 27,794 hostnames and retrieved 843,561 hostnames in total, resulting in 301,219 unique hostnames after de-duplication, of which only 7,723 were repeated from the top million datasets. 134,812 remained after strict filtering for government hostnames. The crawls were completed from the University of Washington between 1st-3rd March 2020. We measured the rate at which the dataset grew from our initial seed list as a result of the crawler. The rate of hostname discovery steadily declines for each level after the 5th level, leaving us with 134,812 unique government hostnames at the end of the crawl.

Number of Govt. Websites	Majestic Million	Cisco Million	Tranco Million
Top 1000 (1K)	56	0	30
Top 10000 (10K)	508	14	373
Top 100000 (100K)	2538	433	2351
Top 1000000 (1M)	12445	9296	12293

Table 5.1: Overlap of Our Government Website Dataset With Public Top Millions

5.2.3 Hostname Search and Manual Curation

Finally, we manually investigated the seed list for each and every country, adding missing websites to ensure inclusion of improperly filtered hostnames, obvious sites from top search engine results, and long-tail countries still having less than 11 total sites after the MTurk tasks. We found these websites via a combination of Google search, manual crawling of seed list links and foreign embassy or non-government travel sites, and careful individual scrutiny for signs of legitimacy as well as impersonation or phishing (to the best of the authors' ability and expertise). This produced a hand-curated list of 596 government hostnames from 62 countries, which we included with the final list of 134,812 filtered unique hostnames, resulting in a total of 135,408. Even after this process, 15 countries remained with less than 11 sites: Chad, Comoros, DRC, Equitorial Guinea, Eritrea, Honduras, Nauru, Niger, North Korea, Palau, Sao Tome and Principe, South Sudan, Togo, and Tuvalu. We also manually added hostnames from Germany, Greenland, Gabon, Denmark, and the Netherlands, which do not use any variation of our expected government domain extensions, as well 14 countries using TLDs such as .com, .org, and .net, to our curated list. We did not crawl these hand-curated hosts with our automated crawler because we could not programmatically confirm linked sites as government-operated without manually visiting and tagging the crawl results. Using the final list of hostnames, we performed measurements between April 22nd and April 26th, 2020.

For the measurements, we performed full TLS and TCP handshakes with the root page of each website and retrieved the certificate chain along with the peer certificate. In case of failures to connect, we performed 3 retries for the hostname by adding the request to the queue. If the host did not return a status 200 code after three attempts, either because the domain name could not be resolved or we could not fetch any content over http or https, we deemed the website "unavailable" and excluded it from further analysis. The results were obtained from a single snapshot. Future work could monitor sites periodically to identify changes in https adoption. [1]

As our authoritative ranking dataset we used the Tranco Million [205], a curated list of top million sites optimized for lower churn and thus more research validity. 12,293 (<10%) of our 135,408 discovered hostnames were present. The small overlap of our generated list and the Tranco million suggest that most of our discovered hostnames likely lie in the long tail of the Internet and outside prior analyses. The overlap with Tranco and other popular top million datasets are presented in Table 5.1. We used *OpenSSL* for validation of certificates and certificate chains downloaded from all of the hosts [258]. To mark a website as valid in our scans, we validate the entire certificate chain. We chose OpenSSL with the default trust store shipped with the Apple Mac operating system [24] imported into the machine over Mozilla's NSS or the Chromium trusted certificate store, since it is the most restrictive and does not include certificates that might be available individually in the browsers' codebases based on their trust with the CA. As a result, our scan shows a small number of certificates as invalid which are valid when using a specific browser or operating system, due to our conservative trust store.

This study was approved by the Institutional Review Board (IRB) and exempted under ID STUDY00009482 by the University of Washington Human Subjects Division.

[1] We identified some inaccuracies due to timeouts from our scanners while measuring the adoption of https for New Zealand, Republic of Congo, Togo, and United Arab Emirates. We performed an additional scan on 9/9/2020 and updated our results.

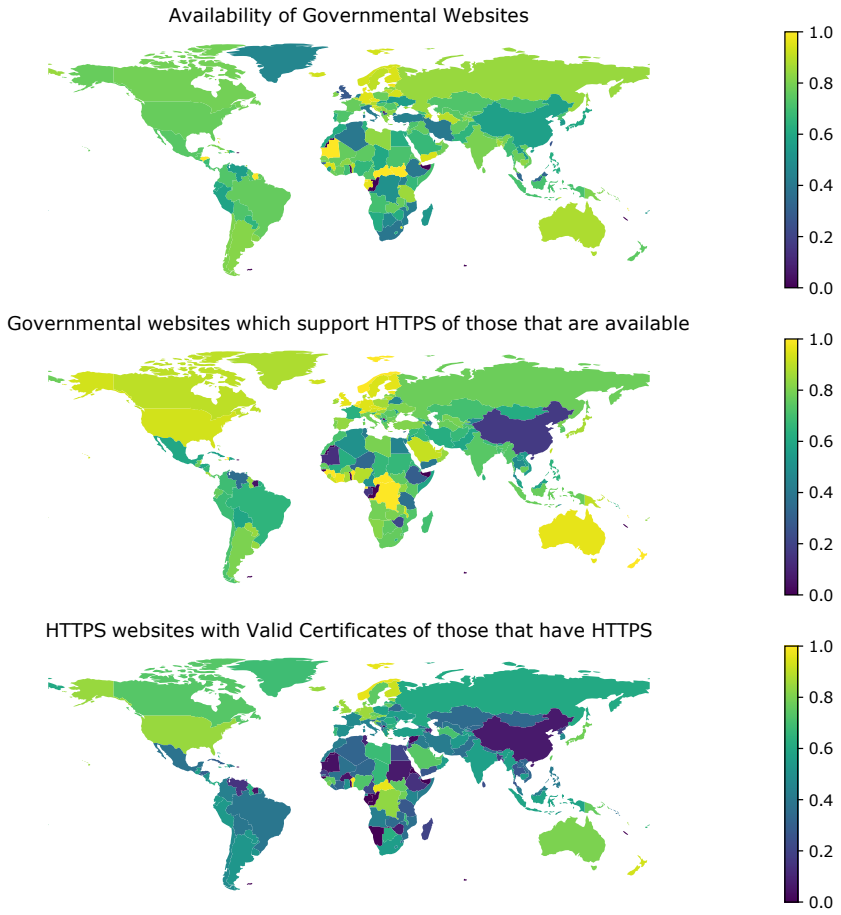


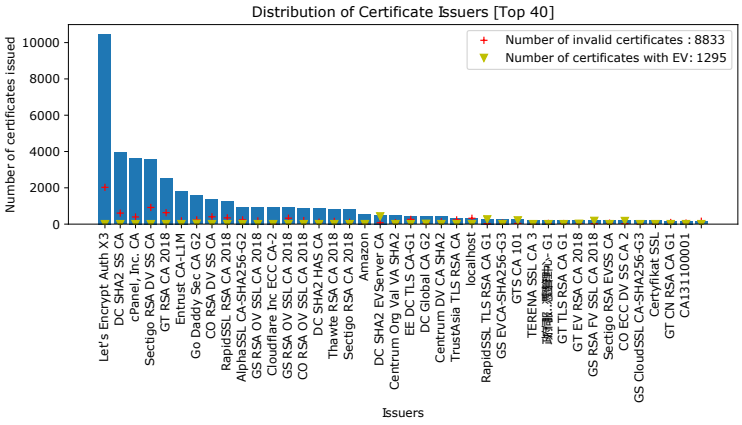
Figure 5.1: Worldwide view of Government Websites

Top: the percentage of government websites from our total list that are available, where the host returns a 200 status code. Middle: the percentage of available sites which support https. Bottom: the percentage of sites that support https which have valid certificates.

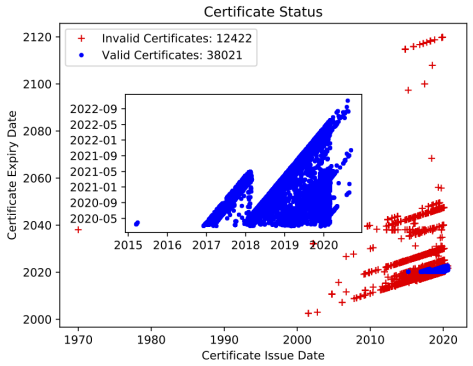
5.3 Results

5.3.1 https Adoption, Use and Issues

Of 135,408 worldwide government hostnames analyzed, 82,152 (60.67%) only support http, while 53,256 (39.33%) serve their content with https. Only 38,033 (28.08%) use https correctly, even



(a) Top 40 Cert Issuers for Government Websites



(b) Certificates by issue and expiry date.

Abbr: COMODO=CO, DigiCert=DC, GlobalSign=GS, GlobalTrust=GT, Encryption Everywhere=EE, "High Assurance Server"=HAS, "Secure Server"=SS

Figure 5.2: Government Certificate Validity and their Issuing CAs

when optimistically including the 4,126 sites that load content on both http and https. We show overall results by country as a choropleth map in Figure 5.1. Within the United States, while a majority of the websites do support https, there are still 1,841 sites (18.45%) that have no https and 1,147 sites (11.49%) serving both http and https.

Most (20.03%) of https enabled government websites worldwide use certificates issued by Let's Encrypt with ≈80% of them being valid. ≈20% invalidity is due to expiry, misconfiguration leading to incorrect certificate usage by the host, or self-signing of certificates. The top 15 CAs used by governments, including Let's Encrypt, do not provide extended validation (EV) certificates. The first major EV certificate issuer, DigiCert, has ~20% invalid certificates for government hostnames, similar to Let's Encrypt. This case suggests EV certificates obtained for a fee may be equally likely to be invalid as *free* CAs. We show a breakdown of the certificate issuers and their number of invalid certificates worldwide in Figure 5.2a.

The top CAs issuing certificates for government hostnames differ by country. For example, the leading certificate issuer in Switzerland is QuoVadis Global SSL ICA G3, while in China it

is Encryption Everywhere DV TLS CA-G1. From a global perspective, Let's Encrypt continues to be the leading CA authority issuing certificates. We expect that this is due to the low cost (free) of certificate issuance and ease of installation with tools like certbot by the Electronic Frontier Foundation (EFF) [123]. Combining valid and invalid certificates, 53,256 websites in our list attempt to serve https web content. Filtering out 2,721 hostnames which have exceptions and other errors, and 92 hostnames without certificate issuer information encoded in their certificate, we analyze the remaining 50,443 hostnames. 19,781 (39.21%) of the sites use a wildcard certificate and 4,486 (22.67%) of these are invalid. We further use the EV policy OIDs in Mozilla's certverifier to check for policy strings corresponding to trusted EV certificates [238], and find 2,145 (4.24%) EV certificate hostnames.

The leading cause for certificate invalidity is *host name mismatch*, contributing to 36.6% of the invalid https certificates. Errors in retrieving *local issuer certificate* and *certificate self-signing* are the next most common. There are instances of government hostnames both using expired certificates and having self-signed certificates in the certificate chain, but this is less than 1% of the hostnames considered. During our scans, 12.7% of the hosts try to negotiate an unsupported SSL protocol (older than SSLv3.0), indicating that the server might be running old unpatched software potentially vulnerable to POODLE [235]. Valid certificates were commonly issued for a fixed duration of 2-3 years as agreed upon by the CAs [69, 70]. Invalid certificates have a much wider spread in duration (see Figure 5.2b). We find 12,422 total invalid certificates due to hostname mismatches, inability to get local issuer certificates, leaf self-signed certificates, and those in the certificate chain along with expired certificates (excluding those causing exceptions). Only 32% of these had a total validity of less than 2 years. 1,746 (14%) were issued for greater than 3 years. 40 certificates had an expiry date 100 years from the year of issue. 617 websites had invalid certificates issued for 10 years, 155 for 20 years, 36 for 30 years, and 1 for 50 years. 1 certificate had an issue date in 1970 (Unix epoch time) expiring in 70 years, likely indicating misconfiguration. 5,372

(43.24%) were issued for a duration in multiples of 365. A detailed breakdown of the certificate invalidity for the websites is shown in Table 5.2.

	Count	**%**
Total websites considered	**135,408**	**100**
➤ Content served on HTTP only	82,152	60.67
➤ Content served on HTTPS	53,256	39.33
➤ Valid HTTPS Certificates	38,033	71.41
➤ Invalid HTTPS Certificates	15,223	28.58
➤ Hostname Mismatch	5,571	36.59
➤ Unable to get local issuer cert	3,732	24.51
➤ Exceptions	2,619	17.20
➤ Unsupported SSL Protocol	1,929	73.65
➤ Timed out	378	14.43
➤ Connection refused	135	5.15
➤ Connection Reset by peer	141	5.38
➤ Wrong SSL Version Number	11	0.42
➤ TLSv1 Alert Internal Error	9	0.34
➤ SSLv3 Alert Handshake Failure	7	0.26
➤ TLSv1 Alert Internal Proto. V.	8	0.30
➤ Self-signed certificate	2014	13.22
➤ Certificate Expired	838	5.50
➤ Self-signed certificate in chain	347	2.27
➤ Others	102	0.67

Table 5.2: Worldwide government sites by https validity and error
All percentages are computed out of the category level directly above it (for example, Unsupported SSL Protocol accounts for 73.65% of Exceptions.)

5.3.2 Host Public Key Pair Algorithms and Reuse

We find a number of patterns relating certificate validity, host public key size, and CA signing algorithm. One-fourth of hosts using RSA with 2048-bit and 4096-bit public keys serve invalid certificates. 520 government hostnames use cryptographically insecure 1024-bit RSA. In the USA, NIST issued a special public document recommending key lengths larger than 1024 with popular tools like OpenSSL being compliant [43]. We also find that RSA key sizes of 3248 bits are generally

misconfigured because of incorrect usage and or 8192 bits due to lack of support in browsers for validating key sizes greater than 4096 bits. We also see an increasing use of elliptic curve (EC) cryptography, dominated by 256-bit keys. However, many certificates still use insecure hash functions such as MD5 or SHA1 during signature validation.

We notice that government websites tend to reuse wildcard certificates across different hostnames belonging to the same government, often incorrectly. One such certificate was shared across 102 hostnames in Bangladesh. However, https was invalid on *all* of these sites because of hostname mismatches; the wildcard certificate was valid for *.portal.gov.bd but was used on all *.gov.bd. In a similar case, the Colombian government used the wildcard certificate for *.micolumbiadigital.gov.co on *.gov.co. Such instances are found in 111 countries, with the top five violators being Bangladesh (2 certificates incorrectly used across 138 hostnames), Colombia (3 certificates incorrectly used across 125 hostnames), China (8 certificates incorrectly used across 107 hostnames), Dominica (1 certificate incorrectly used across 28 hostnames), and Vietnam (3 certificates incorrectly used across 21 hostnames). Unlike cases where a single certificate is shared across different hostnames in one country, we also see instances of public key and single-certificate reuse by *different* governments.

We found 58 government hostnames of 24 countries using the same certificate. 154 certificates were reused across 1,390 hostnames, with 108 certificates reused by 2 countries, 19 by 3 countries, 11 by 4 countries, and 1 by 24 countries. The most-reused certificates are invalid self-signed localhost certificates with the same set of public keys. 210 (15.1%) of these hostnames use self-signed certificates with no chain of trust, while 648 (46.6%) of the incorrectly reused ones are invalid due to hostname mismatches. This incorrect usage points to a troubling possibility that all the servers share the same private key. A malicious user with the key could observe TLS connections to a target server using the same certificate and decrypt communications with any clients who have added an exception to the invalid certificate. Valid reused certificates are wild

card certificates being hosted by the same government. We do not find any instances of valid public key reuse across country governments.

5.3.3 Comparison with Non-Govt. Sites

Given the positive effects of public and commercial pressure on https adoption, we expected website popularity ranking and use of valid https to be correlated. This complicates an apples-to-apples comparison between government and non-government sites as our list includes mostly government sites outside the top millions (90.9%), for which there are no rankings. Thus, we restrict our comparison to the subset of our government hostnames present in the Tranco million dataset (12,293 of our 135,139 hostnames), comparing https validity while accounting for relative rank.

We compare https in these top government websites (mean rank: 396,427, σ: 285,611) with [1] 12,000 random, uniformly sampled top million non-government hostnames (mean rank: 499,206, σ: 286,907) and [2] 12,000 sampled top million non-government hostnames closely matching the rank distribution as the government hostnames (mean rank: 402,676, σ: 288,942). For sampling dataset [2] of non-government hostnames, we first divide the top million into (N=50) buckets by rank, and count the number of government hostnames in each bucket, ensuring each contains at least 100 government hostnames. We then uniformly sample an equal number of non-government hostnames in each bucket to match the number of government hostnames. Figure 5.3 compares these 3 sets with linear regressions on https validity by rank, with 95% confidence interval bands.

Though ranking does have an effect, overall valid https use in government websites in the top million is similar to results in the long tail dataset, at ~30%. Meanwhile, the top 12,000 non-government websites have >70% valid https while the two non-government sets we sampled have ≈55%, indicating that even top government websites perform worse than most other top million sites. We expect that these results likely remain consistent in the long tail of the Internet.

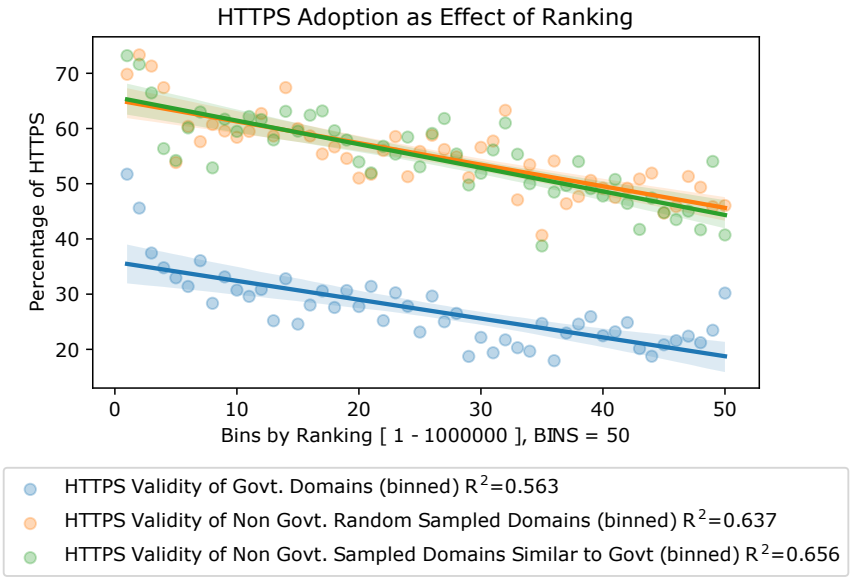


Figure 5.3: Valid https rate plotted by top million rank

Percentage of valid https of Government and Non Government websites in the Tranco top million by ranking, with sites grouped into 50 bins. Plotted data are the 12K government hostnames listed (blue), randomly sampled non-government hostnames (orange), and randomly sampled non-government hostnames with a rank distribution matching the government sites (green). Linear models show a trend of decreasing https with rank for all sets, with worse https adoption for government sites.

Figure 5.3 shows this disparity and indicates that the probability of having a valid https certificate reduces as ranking worsens.

5.4 Impact of Notification and Disclosure

Government domain name registrations are typically handled by a separate registrar for each government and expected to meet stringent verification requirements. For example, on March 5, 2020 the US (dotgov) registrar made it mandatory to obtain notarized signatures on authorization letters when requesting a .gov hostname [138] due to impersonation attacks [192].

As a part of our analysis, we generated reports per country of potentially vulnerable hostnames, including invalid https, failed upgrades of http to https, and unreachable hostnames which were still linked from other pages that we discovered during the analysis. We emailed the respective country government domain registrars (performing *whois* queries on the country registrars to find listed technical contacts), included their vulnerable hostnames as a file attachment and requested contact information for the domain owners or appropriate forwarding of our reports. We sent emails to 182 countries, since 9 (Angola, Benin, Democratic Republic of Congo, Estonia, Guinea, Netherlands, Norway, Switzerland, and Vanatu) countries had https for every detected hostname, and at the time of disclosure we still had no hostnames for 7 countries due to our conservative filtering. 175 emails were delivered, and 7 bounced. We retried the 7 countries by emailing the listed administrative contact, of which 4 emails failed again. 6 registrars sent an automated message acknowledging the receipt of our email. Some of the countries with valid https for all hostnames had very few in total (≈30); an in-depth analysis of these countries may be needed to clarify the correctness of this result.

Responses were surprisingly positive. 39 domain registrars were supportive: 3 (Brazil, Lebanon, and Liberia) provided us with the necessary contact information, 13 of them (Austria, Bosnia & Herzegovina, Burundi, Cayman Islands, Columbia, Lithuania, Netherlands, Nigeria, Rwanda, Sri Lanka, Tanzania, Tonga, and Ukraine) re-directed our emails to the corresponding government authority or a responsible person who was the intended recipient, and 2 (Japan, and Norway) responded mentioning that they could not provide us with contact information that isn't publicly available in the whois and suggested that we query their *whois* servers for the information. The registrar of one country responded negatively, saying *"We are not interested"*.

Previous work by Stock *et al.* showed that transmission of vulnerability reports to actual domain owners had very limited impact on successful resolution, and only resulted in a very small (~5.8%) number of emails being actually received [323]. In our study, 22% of the country

government domain registrars / CERT authorities proactively replied to our messages and have begun taking the necessary steps to fix their certificates. We believe that for government domains, the registrar, who might represent a government body, may have a higher incentive to respond to such reports than individual developers and be in a position to make a meaningful change.

Two months after notification, we scanned the 15,179 government websites with previously invalid https to understand notification effectiveness. 1,572 of these were unreachable and seem to have been removed, while 1,263 websites had fixed the certificate invalidity issues. 12,344 sites continue to serve content with invalid certificates. Assuming that newly unavailable websites (no longer returning a 200 status) have been removed on purpose by webmasters and considering this a fix, we optimistically estimate improvement at 18.7%; otherwise, the improvement is only 8.3%.

Of the 47,458 sites unreachable in our original scan and thus not considered in our list of 135,139 sites, we notice that 38,077 continue to be unreachable, while 2,850 (6%) sites now serve content using an invalid certificate, and 6,531 (13.76%) sites serve content with a valid certificate. 950 (1.15%) websites which previously served http-only traffic now serve valid https traffic, while 1,523 (1.85%) websites serve content with an invalid certificate, and the remaining (96.9%) continue to use only http. Preliminary findings indicate that notifications and disclosure do have a small positive effect, with 62 countries showing at least a 10% improvement in valid https and 7 countries (Bahrain, Burkina Faso, Cuba, Honduras, Portugal, Libya, and Vietnam) showing improvement above 40%. Since we do not perform a full re-scan of all hostnames and only measure changes in sites which were previously invalid, we cannot measure deterioration of websites. The United States government issued a statement after our disclosure mandating HSTS preloading for ".gov" websites by September 1, 2020 [112].

5.5 Conclusion

Many (≈72%) government sites do not still use https, either due to lack of TLS infrastructure or a large variety of certificate errors. The implicit trust placed by citizens and users on the services and the information provided by such infrastructure which lies typically hidden in the long tail of the Internet, raise significant concerns for security of the content and its trustworthiness. User activity to these websites due to their insecure nature leave them vulnerable to attackers – both passive and active. With the emergence of automated certificate issuance and renewal processes, government websites which do not use official government related domain names risk potential domain spoofing and phishing attacks. Additionally, it might be valuable for individual country governments to understand risks due to 3rd party certificate authorities issuing their certificates leaving them at risk of compelled certificate creation attacks. The increased usage of https improves user security and keeps their interactions private and confidential from eavesdropping adversaries.

Chapter 6

Private, Secure, and Transparent Internet Interactions

With the ability to securely connect to the Internet through a decentralized set of carrier networks, users now interact with various components and services on the Internet. Despite the Internet's humble decentralized beginnings, the increasing popularity of the cloud providers and few large infrastructure providers have reshaped the Internet today.

Today, a large portion of the Internet traffic flows through the networks owned and managed by a few large organizations rather than through multiple smaller transit providers [313]. This is due to the extensive investments made by cloud and content delivery network (CDN) companies in infrastructure to improve performance by moving infrastructure closer to the users. So, while the Internet infrastructure is distributed globally and resilient, it is organizationally centralized. As a result despite the usage of technologies such as Transport Layer Security (TLS) which are popular ways to enable secure communication between users and the content on the Internet [337], we trust a handful of organizations globally. The trend of co-location of content with infrastructure has opened up opportunities for improving user experience, performance, and resilience, but comes with significant privacy and security concerns. Users of the Internet still perceive the Internet today as the decentralized network it was originally designed to be, despite heavy co-location in the underlying infrastructure.

6.1 Protecting User Privacy during DNS Interactions

Once connected to the Internet, user's through their network traffic interact with hidden infrastructure such as a sequence of routers along their network path to the intended destination as well as critical infrastructure such as the domain name system. The Internet's Domain Name System (DNS) responds to client's human memorable hostname queries with corresponding IP addresses and other associated records. Once authenticated with their services, access providers like cellular networks or ISPs manage infrastructure that automatically sends the required network configurations containing their IP address, DNS configuration and Internet gateway information to the user devices. Typically, the DNS resolvers are operated by the access providers themselves within the network perimeter.

By default, DNS messages are transmitted in cleartext over the User Datagram Protocol (UDP) on port 53, hereafter referred to as Do53. As a result, Do53 is vulnerable to eavesdropping and modification by both well-intentioned and malicious third parties. Not only do these risks compromise privacy, but they can result in denial of service (DoS) and injection attacks [359]. Compared to the broader Internet and web ecosystem, wherein traffic is increasingly safeguarded by HTTPS, Do53 lags behind and remains as a weak facet of a secure Internet. Secure variants of DNS have recently been introduced to fill that gap. DNS-over-TLS (DoT) [160] and DNS-over-HTTPS (DoH) [156] are now widely supported by browsers and increasingly supported by operating systems. In DoT and DoH, the *transport* of DNS messages between client stub resolvers and upstream recursive resolvers is encrypted over the TLS channel. However, DoT and DoH are supported by only a small number of providers relative to the wider resolver population.

Although DNS benefits from auto-configuration of local services in DHCP [113], there exists no equivalent beyond the local network, nor does there exist a discovery mechanism of any kind for DNS. This has exposed two challenges with respect to deployment and privacy. First, the majority of clients leak or expose their information to on-lookers. For example, at time of

writing, we observe from a large recursive DNS resolver that clients transmit 92% of their queries (254 billion/day) using the default cleartext Do53 protocol [269], and therefore leak their own information. In the absence of wider deployment, discovery, and auto-configuration supports, the only way to use secure DNS is by manual setup.

Due to the implicit trusted nature of the DNS resolvers configured by the access providers to the user, the user in addition to the access to the network, also trusts their service provider with their browsing pattern over the Internet – raising serious privacy concerns. hile any individual website or online service can associate requests (and user data) with clients IP addresses for their own service, DNS resolvers' ability to observe queries makes it possible to link *all* client activity. One resolution to this issue is to rely on trust-based policies or legal agreements. Public resolvers, for example, publish their own privacy policies; but these are non-uniform, and require domain expertise or language proficiency to understand. Similarly, contractual agreements such as in Mozilla's Trusted Recursive Resolver (TRR) program [103, 241] can define and restrict data retention, aggregation, sale or transfer. These mechanisms, however, are hard to validate and lack any technical means of enforcement.

A better approach would be to modify DNS itself to disassociate query contents from IP addresses in a way that makes them un-linkable. Oblivious DNS (ODNS), for example, makes resolvers blind to both client IPs and their queries [301]. In ODNS, stub resolvers encrypt client queries into a new query for an .odns top level domain (TLD). Resolvers are then forced to direct the query to an ODNS server that presents as authoritative; the ODNS authoritative server then decrypts and resolves the query as a normal resolver, before receiving and returning an encrypted response. ODNS works in Do53, but requires registration of a new tld, changes the semantics of authoritative service, and redirects all queries to their authoritative name servers instead of local and performant recursive resolvers.

6.1.1 Existing Encrypted DNS Protocols and Privacy

Efforts to secure DNS via message or channel encryption include DNS-Over-TLS (DoT) and DNS-Over-HTTPS (DoH) [156], a precursor in T-DNS [359] and, separately, DNSCrypt [132]. Client support for DoH is increasingly available among web browsers [33, 103, 271], mobile clients [106, 107, 189], and operating systems [207]. This has been attributed to the availability of DoH support in public recursive resolvers offered by Cloudflare and NextDNS, with their integration into Firefox as Trusted Recursive Resolvers [60, 103, 213]. Organizations that operate public DoH resolvers publish and adhere to their own privacy policies. These policies may prohibit user tracking, and state strict data use and retention practices [108, 139].

Other organizations may aggregate DNS traffic patterns into permanent logs that omit identifiable information like IP addresses [76, 338]. Reasons may include telemetry for debugging in production systems, improving response time performance, or for identifying browsing trends by geography [108, 136]. Such practices are important because IP addresses are often regarded as personally identifiable or linkable information [130, 223].

In DoT and DoH the network channel is encrypted, and serves as a secure transport for the plaintext messages that are transmitted. Conversely, DNSCrypt [132] and its predecessor DNSCurve [53] encrypt the DNS message before its transmission over TCP or UDP. A client connecting to a DNSCrypt resolver receives a public set of signed certificates that is verified by the client using a known provider public key. Each certificate contains a short-term resolver public key. Messages are encrypted with a shared key generated from the resolver public key, the client secret key, and an agreed key exchange algorithm defined in the certificate. The shared key is used to encrypt subsequent queries using an authenticated encryption algorithm.

Neither channel, nor message-based encrypted DNS protocols address privacy risks at the resolver, where client address and queries can be logged, stored, and potentially profiled or transferred. Anonymous DNSCrypt addresses this issue by introducing public non-logging proxies

that forward traffic between clients and the intended DNSCrypt resolver [104, 272]. However, since the traffic is between the proxy and resolver, an adversary that is incident to a proxy's ingress and egress link could link queries to clients.

Prior proposals to address privacy included Privacy Information Retrieval (PIR) techniques. *Range queries* use random noise and PIR techniques during execution of DNS queries [357, 358]. Doing so provided confidentiality, integrity, and privacy when used in conjunction with DNSSEC. Later evaluation of these techniques revealed that substantial changes to DNS servers and clients were necessary to make it feasible, as well as that attackers with control of the channel could infer and forge queries [73]. Cryptographic mixes or mix cascades have also been leveraged to anonymize user traffic [54, 127], which is a model widely adopted in Tor. Additional attempts at privacy consist of broadcasting a desired query among a set of other decoy queries to thwart profiling attempts [127]. Doing so has practical limitations due to large bandwidth usage and long-tailed distribution of queries. These limitations led to a hybrid of mix cascades and broadcasts [127], albeit with degraded performance due to increased page load and DNS response times.

DNS privacy may also be achieved using DoH over Tor (DoHoT), in which the encrypted DNS channel is routed over the Tor anonymous network [87, 244]. Doing so closes the linkability gap of Anonymous DNSCrypt, but also incurs substantial performance penalties [244]. However, recent work by Muffet indicates performance penalties due to the usage of Tor might not impact some users significantly and could go unnoticed by users because latency is only a small fraction of the user experience and value proposition for choosing a DNS protocol [243]. While Tor provides both anonymity and privacy guarantees, Tor nodes can be actively censored or blocked throughout the Internet [332, 354]; traffic can be subject to DNS fingerprinting attacks [140]; and operators of exit nodes may face legal liabilities [151, 218].

6.1.2 Privacy and Regulatory Considerations

DNS service provision and traffic also invokes a number of regulatory, economic, and even philosophical considerations that, while non-technical and beyond scope, are worthwhile touching upon. Among them is the manner in which a local DNS service is automatically configured by upstream ISPs, and the implications that follow [59]. We stress that any configuration mechanism is necessitated by the absence of a DNS discovery mechanism. Beyond configuration, evidence suggests that even seemingly benign activities, such as monetization of only the error traffic in DNS [348], may have unanticipated consequences. The introduction of encrypted DNS protocol services offered by third party public DNS service providers makes it challenging for ISPs to enforce filters, or block sites using DNS.

Users and clients may instead configure their DNS to point to any of a small number of large open resolvers that promise performance and consistent quality of service across networks. The performance improvements are accompanied by an implicit shift in trust to the open resolver, and their operators to publish privacy policies and conduct audits. A trusted policy may be sufficient, but is unable to protect in all circumstances that include human error or system compromise.

Large scale measurements of encrypted DNS protocols show that adoption has been increasing [213]. Additional measurements reveal that DoH provides security with no significant impact on page load times [60]. Large scale evaluations from home networks indicate that latency becomes the performance bottleneck as broadband access speeds increase, making metrics like DNS response time and time to first byte more important [324]. Additional measurements suggest that the performance of DoT and DoH vary with the choice of public resolver [158].

6.1.3 Oblivious DNS over HTTPS (ODoH)

A recent standard at the Internet Engineering Task Force (IETF) was proposed introducing ODoH. We rigorously analyze, evaluate, and implement this new DNS protocol – that with minimal

changes allows users to improve their privacy while interacting with DNS resolvers and builds on the concepts of ODNS to decouple queries from IP addresses to ensure privacy. The broad mandate of Oblivious DNS over HTTPS (ODoH) is to address the remaining privacy concerns of encrypted protocols like DoH/DoT i.e. to prevent recursive resolvers from being able to link client IP addresses to their queries thus reducing the implicit trust placed in encrypted DNS services.

Features and Implementation

The design of ODoH is similar to that of DNS over HTTPS (DoH). It differs with the addition of an intermediate proxy node that forwards queries and responses between a client and target in an 'oblivious' manner. The oblivious property derives from two attributes. First, connections to and from the proxy use HTTPS to secure the message transmissions from eavesdroppers. However, a proxied-variant of DoH, alone, would expose the query to the proxy. For this reason ODoH secures the payload from the proxy with an additional layer of end-to-end encryption between the client and target for a query. The combination of HTTPS, with the intermediate proxy, and end-to-end encryption ensures that *only* the client knows both its identity (IP address), queries made, and responses intended for it. Beyond this, ODoH achieves the following properties: (1) Proxies know client IP addresses (i.e. identities) but cannot see actual queries and responses, and (2) Targets and entities upstream involved in resolution know the query, but only see the IP address of the proxy, hiding client IP information.

Protocol Description

ODoH participants, as well as the protocol, are depicted in Figure 6.1. We describe the design from the view of each of the ODoH components: (i) Clients that communicate using a stub resolver; (ii) an Oblivious Proxy that transmits messages over HTTPS between the client stub and (iii) the Oblivious Target that encrypts and decrypts messages between client and DNS resolver.

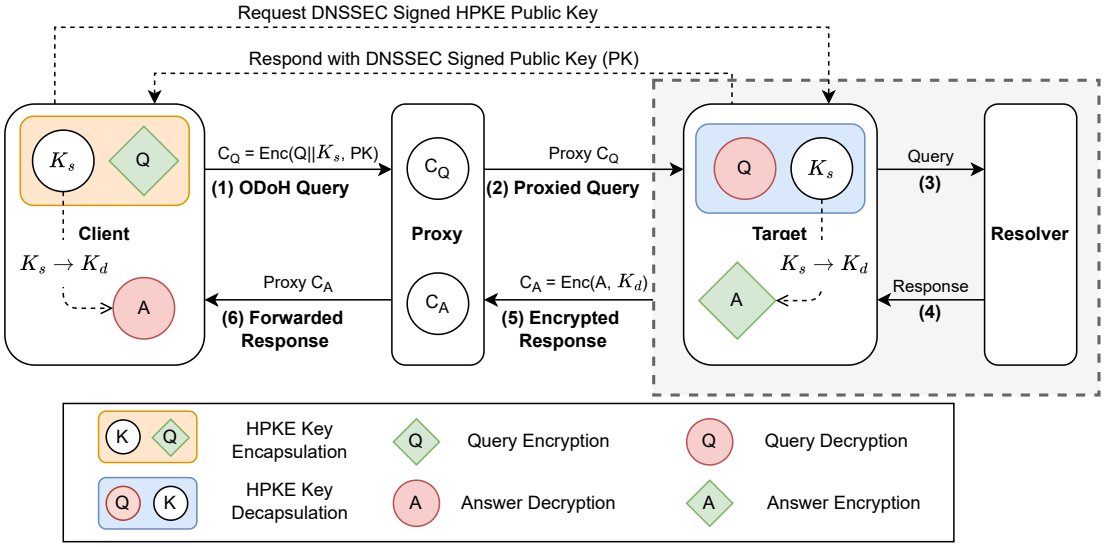


Figure 6.1: End-to-End Execution of the ODoH Protocol

Clients (or stub resolvers): The protocol begins at client's stub resolver, responsible for encrypting client queries and decrypting target responses. Clients are free to use one or more proxies, as well as one or more targets. Pairs of proxy and target may be selected per-query, so long as there is no collusion between them, i.e. they are operated by separate entities. The client encapsulates an ephemeral fixed size shared secret symmetric key K_s with the DNS query Q which is encrypted using the public key of the target PK, resulting in the encrypted message C_Q that is sent to the chosen proxy as shown in step (1) in Figure 6.1.

Oblivious Proxies: The ODoH path consists of two HTTPS connections that meet at an Oblivious Proxy which acts as server to the client, and as a client to target servers. On path from client to target, the proxy preserves the encrypted query C_Q, removes client IP address information, then forwards the request to the Oblivious Target indicated by the client's message as shown in step (2) in Figure 6.1. Responses from the target are similarly returned to the client that originated the query as shown in step (6) in Figure 6.1.

Oblivious Targets: DNS messages in ODoH are secured with Hybrid Public Key Encryption (HPKE) [45, 46]. Targets use keying material derived from a query's HPKE encryption context to encrypt the response. Encrypted queries (C_Q) from the client arrive to the target via the proxy. The target decrypts and decapsulates the message received resulting in the query Q the shared secret K_s. The target resolves the query, computes a key K_d from the corresponding Key Derivation Function (KDF) based on the shared secret K_s, and then encrypts and returns the answer A as an encrypted response C_A via the same proxy. These communications are shown as steps 3,4, and 5 in Figure 6.1. The Oblivious Target publishes a public key which the clients securely retrieve and use to encrypt their DNS queries before the protocol is run and is shown by the two dashed lines between the client and target in Figure 6.1. We note that the target is defined as being independent of the resolver, but is expected to be coupled with the recursive resolver to improve performance by avoiding two additional network messages. Our implementation and evaluation focus on co-location, and we anticipate that ODoH target features could be implemented within existing recursive resolvers.

End-to-end Message Flow: The overall end-to-end flow of an ODoH query across clients, proxies, and targets is shown in Figure 6.1. The dashed box with a grey background that surrounds target and resolver indicates their possible coupling or co-location, while respecting their independence according to the definition. Co-location trades isolation for performance, without loss of privacy. A target co-located with the resolver has no need for messages (3) and (4) shown in Figure 6.1. The split of responsibility between non colluding entities reduces the implicit trust that is placed on the resolver while providing abilities for the client to identify when participants they communicate attempt to be malicious allowing user clients to take corrective measures. In the ODoH protocol, a proxy does not learn the contents of the query from the client or the response from the target, and similarly, a target resolver does not learn the identity of the client – improving privacy of the underlying DNS interactions.

Discovery, Verification, and Key Distribution: Strictly speaking, DNS in all its forms has no discovery service. Aside from manual configuration, conventional Do53 is traditionally configured (ultimately by ISPs) with DHCP. DNS over TLS (DoT) and HTTPS (DoH) are used by client stub resolvers by opportunistically probing the DNS server indicated through the DHCP configuration for support for either protocol [189]. In this respect DoT/H and ODoH are equivalent: similar probing based approaches to discover protocol support by the resolver can be reused. As a result, in the absence of DHCP support and until a DNS discovery service emerges, options for ODoH configuration are limited to hostnames listed on centralized and trusted lists that could additionally be bootstrapped into browsers or operating systems which might choose to support the protocol or through manual explicit configuration.

Once a proxy or target is known, each client stub resolver requests a DNSSEC signed resource record set (HTTPSVC, SVC, or similar special fields [304]), where the ODoH public key for the server is published. Once the signature is validated, the stub resolver adds the oblivious target to the list of verified targets and stores the public key necessary for communication. These records can be periodically retrieved and validated by the client stub resolver to check for updates. A browser or client then directs a special lookup to the resolver for odoh.test, which returns the necessary keys for validation and indicates support for ODoH. This mirrors doh.test to check for DoH support [239] – an intentional choice; reducing barriers to adoption, and deployment.

Formal Analysis & Replay Attack Mitigation

We provide a formal analysis to prove that the ODoH protocol provides client privacy. Our analysis consists of a symbolic model generated for the Tamarin theorem prover [300][1].

Our model is kept simple, yet precise, and focuses on the security and privacy of the DNS request-response exchange in ODoH. Thus, we intentionally ignore longitudinal attacks and simple

[1]Tamarin has also been used successfully to analyze security properties of TLS 1.3, 5G, & other protocols [97, 98].

correlation-type attacks such as only one client using a given proxy are either trivially simple or impractical. Similarly, the integrity of resolver responses or their ability to de-anonymize clients is beyond scope of the protocol, and our analysis. ODoH makes no claims of resolver response integrity. A malicious target or resolver could craft answers in such a way to de-anonymize users. This is also discussed in the ODoH protocol specification.

Our model captures cryptographic protocol details, for example, HPKE-based [45] query encryption, and AEAD-based response encryption. Underlying cryptographic algorithms such as hashes are otherwise assumed ideal. Other aspects of ODoH are both impractical and unnecessary. For example, the complexity of TLS 1.3 needs hundreds of thousands of steps to prove [98], so we substitute two simplifications. First, we assume that the client and proxy can establish an authenticated shared key, which is the end-goal of TLS. This assumption relies on TLS being implemented, instantiated, and deployed in a secure way. This is reasonable, given the scale of and complexity of attack to break this key in practice [14, 249]. Moreover, we do not model precise wire-format details of ODoH, and only capture message components in an idealized way since the wire format should not affect protocol security.

As an additional simplification for the analysis, we omit the TLS layer between the proxy and target. Specifically, because we give the attacker the ability to compromise TLS sessions secrets, this simplification is equivalent to the attacker always doing so. We assert that this is a fault-preserving simplification [161], wherein any attack that succeeds on the true protocol also exists in the simplified model. Thus, a proof that the simplified model is secure also implies that ODoH as specified and deployed is secure.

While ODoH provides strong cryptographic security properties for individual queries, it has limited defences against correlation attacks. Backes *et al.* formalise this as *Sender Anonymity*, i.e., whether an attacker can identify the sender of a message with some non-negligible probability [31]. As discussed in Das et al. [99, 100], resisting correlation attacks in the presence of an adversary

that can observe both endpoints requires added latency or bandwidth overhead. In particular, they provide a lower bound on the latency required to achieve strong anonymity of this type. Moreover, the amount of latency introduced has to grow as the number of participants grows [99, Theorem. 8]. DNS has near-real-time performance requirements, so latency overhead is particularly problematic. We see this later in the comparison with DoHoT, where the use of Tor incurs a performance loss from longer circuits at the potential gain of stronger protection against correlation attacks in practice. We thus settle for a weaker form of anonymity, where an attacker cannot identify the contents of a single session beyond statistical inference. Note, importantly, that an attacker which has compromised the target is perhaps less likely capable of observing traffic between the proxy and the client. The results of Das et al. [99, 100] do not preclude there being any real-world gain in privacy. ODoH protects against attackers who can see large portions of the network, and because of its cryptographic guarantees even an attacker that can link the client and the target still needs to compromise the target to learn the query. DoHoT provides more anonymity protection, although it still falls short of strong anonymity [99, Table I], but at a cost of very substantial latency. ODoH provides a good balance between privacy protection in the real world and usable performance. Its simplicity of design and architecture enable it to be deployed widely, and its relative performance compared to protocols with stronger guarantees encourages adoption.

The adversary in our analysis is an extended Dolev-Yao attacker [111], with the standard ability to create, drop, and modify messages. We additionally give the adversary the ability to compromise TLS sessions and the target server's long-term key (LTK). Finally we give the attacker the ability to compromise the security of AEADs if keys and nonces are reused. Obviously against an adversary such as this, ODoH is not secure. However, placing careful limits on the attackers behaviour yields tight security bounds.

Security Properties: Collusion is modeled by allowing the adversary to corrupt proxies and targets at will, and is achieved if the adversary is able to compromise both for any single query.

An adversary that does not compromise both effectively models a non-colluding proxy and target. With the ODoH protocol, we prove the following statement:

Core Lemma *An adversary is unable to associate a connection between client and proxy with the corresponding query unless both the proxy and target are compromised.*

Our model also proves that if the adversary controls *all but one* servers, client privacy is preserved when its query is handled by the one uncorrupted server – regardless of the server being a proxy or a target. We also prove that the attacker can only learn the response by compromising the target or by having advance knowledge of the query.

Two additional lemmas are proved to ensure correctness: (i) the protocol runs to completion; and (ii) the client and target agree on the target's identity, the client and target's public keys, the query, and the response. The agreement property is an authentication lemma that proves either that the adversary corrupted the target, or the response was provided by the expected target. This ensures that there are no misdirection or message swap attacks, wherein the adversary reroutes messages to confuse the client about which target responded.

Replay Attacks: Whilst performing our analysis, we discovered a replay attack in older versions of the ODoH protocol. In particular, malicious proxies can log and replay the encrypted client queries and force AEAD key and nonce reuse for the encrypted responses. This was because ODoH previously derived the target response key and nonce entirely from the client's query. If the target's response changed, as is expected with DNS, the encrypted message also changes. This allows the proxy to learn the XOR of two different response plaintexts, which breaks semantic security of response encryption. After discovering this issue, the protocol was modified such that targets include a fresh nonce in the response encryption key schedule. This nonce is sent alongside the ciphertext, allowing clients to derive the same secrets and decrypt the response. We were able to model both the attack and fix in Tamarin, proving that the fix prevents the attack.

6.1.4 Implementation & Measurement Methodology

We implement the protocol presented in §6.1.3 and perform microbenchmarks and multi-point wide-area measurements to evaluate ODoH and present comparisons to other secure and anonymous DNS variants.

Implementation and Microbenchmarks

We implemented two interoperable variants the Oblivious Target and Proxy using both Go and Rust. The implementations were deployed and tested on Google Cloud using Google App Engine [83] and a serverless platform - Cloudflare workers [86]. We also implemented a client in Go with a command line interface similar to dig. The client performs ODoH queries to a chosen Oblivious Proxy and Oblivious Target. The client can optionally select proxy and target using a latency-based heuristic. For performance evaluation, a benchmarking submodule was also implemented within the client. Our benchmarking tool launches C client processes, each performing N queries chosen randomly from the Tranco top million dataset [267] at a rate of R DNS requests per minute.

Both the client and proxy can be configured to reuse HTTPS connections to avoid the extra costs of TCP and TLS handshakes across queries. To accelerate adoption we also integrated ODoH protocol support into popular open source client stub resolvers that support DoH or encrypted DNS protocols like DNSCrypt. We use these stub resolver implementations to perform page load time measurements. Our ODoH implementations across clients, proxy, and target domains are publicly available and have also been open-sourced [252].

Cryptographic Compute Overheads

The default ciphersuite in our implementations is the HPKE ciphersuite as published in the IETF draft [45], consisting of the DHKEM(X25519, HKDF-SHA256) Key Exchange Mechanism (KEM), SHA256 based Key Derivation Function (KDF), and AES-128-GCM based Authenticated

Microbenchmark	Type	P99 **Overhead**
Query encryption time	HPKE	360μs
Query decryption time	HPKE	246μs
Query size (Type A)	HPKE	148 bytes
Answer size (Type A)	AES-128-GCM	98 bytes

Table 6.1: 99th percentile (P99) Overheads incurred by the default cryptographic suite in our HPKE implementations. HPKE uses the DHKEM(X25519, HKDF-SHA256), HKDF-SHA256, and AES-128-GCM ciphersuite

Encryption (AEAD) algorithms. This ciphersuite is performant and widely supported in production environments. We ran our microbenchmarks on a 1 Intel Xeon 2.0 GHz CPU core with 3.75 GB of memory in a Google Compute Engine virtual machine. The compute overheads incurred by HPKE were evaluated by running microbenchmarks on the hosted client instances at sub-microsecond granularity. A set of 10,000 domains were randomly selected for this test from the Tranco million dataset [267]. For each domain the encryption and decryption operations are executed sequentially, i.e. encrypting then decrypting the query before encrypting then decrypting the response.

The main observations are summarized in Table 6.1, with entries at the 99^{th} percentile. DNS query encryption using the most performant ciphersuite (DHKEM(X25519, HKDF-SHA256), HKDF-SHA256, and AES-128-GCM) takes 360μs, while the decryption of such an encrypted query takes 246μs. The protocol also supports other cipher suites involving X448, P256, and P512 KEM; SHA384 and SHA512 KDF; and AES-256-GCM and ChaCha20Poly1305 AEADs. In some cases P256 curve might be preferred over the X25519 KEM for NIST and FIPS standards compliance [273]. For the symmetric key operations AES performs better than ChaCha20Poly1305 due to hardware acceleration in some platforms [183]. The slowest encryption times at the 99^{th} percentile is 430μs using the DHKEM(X25519, HKDF-SHA384) KEM with different possible permutations of the supported KDFs and AEADs; decryption is 713μs when using the SHA-384 and AES-256-GCM.

Our results reveal least compute overhead with HPKE query encryption and AEAD response

encryption. However, all the computation overheads in the lifecycle of an ODoH query, i.e. from query encryption at the client to decryption of the response, take less than 1ms.

Cryptographic Network Overheads

Compared to baseline cleartext DNS messages, HPKE encryption increases the size of the query message on the wire by 4x, and the and the size of the response by 1.2x. This microbenchmark was performed at the client stub resolver. A random sample of 10,000 unique domains were used to serialize cleartext DNS and ODoH query messages. The average encrypted ODoH query size for type A queries is 314% greater at 140.8 bytes. The ODoH query response sizes for the same domains were closer in size to their cleartext counterparts, with a size increase averaging 22.37% to 87.5 bytes. Note that ODoH messages must encapsulate the client public key used to encrypt the query in addition to the query, itself, and appended to an integrity hash.

Measurement Setup

Our measurement setup consisted of clients, proxies, and targets deployed across the USA, with additional locations in Canada and Brazil. We deployed 9 client VMs into US Google Cloud data centers. All VMs had 1 Intel Xeon 2.0 GHz CPU core with 3.75 GB of memory on the x86_64 architecture [84], with an average bandwidth of 480 Mbit/s. For each client, corresponding proxy and target instances were geo-replicated using serverless platforms to the nearest datacenter on the Cloudflare network. An additional proxy and a target were deployed using Google app engine to a Google Cloud datacenter on the west coast of the USA. During the experiments, 10 query processes were launched by each of the 9 client stub resolvers. Each process mimics real client behaviour by making 200 ODoH queries at a rate of 15 requests per minute, or ~21000 DNS requests/day. Parameter values were drawn from Wireshark packet traces of real clients [200, 277]. From each target, client queries were forwarded deterministically ($\mathrm{lastByte} \bmod 3$) to open

Protocol	Query Path	Security	Privacy	Description
DoH - DNS over HTTPS [156]	C → R	Yes	No*	HTTPS based DNS over HTTPS
pDoH - Proxied DoH	C → P → R	Yes	No	Proxy performs DoH query
Cleartext ODoH	C → P → T → R	Yes	No	DoH query is proxied to a Target which performs DoH query
ODoH - Oblivious DoH [185]	C → P → T → R	Yes	Yes	Oblivious DoH protocol
Co-located ODoH	C → P → (T+R)	Yes	Yes	ODoH with Colocated Target and Resolver
DNSCrypt [132]	C → R	Yes	No*	UDP-based encrypted DNS
Anonymous DNSCrypt [104]	C → P → R	Yes	Yes	UDP based, Proxy routes client query
DoHoT - DoH over Tor [87, 244]	C → Tor → R	Yes	Yes	DoH over Tor Network

Table 6.2: The set of *Secure* DNS platforms evaluated alongside ODoH with request path, attributes and description, for reference. Entries with citations are known and publicly available; entries without citations represent one-step at a time architectural changes towards ODoH and beyond. In the 'Query Path' column, C - Client, P - Proxy, R - Resolver, T - Target.

*-Guarantees of privacy depend on the resolver being used & relies on legal agreements or privacy policies.

DoH resolvers (Cloudflare DNS, Google DNS and Quad9) to distribute load and mitigate against possible upstream effects of single-resolver selection.

Identical sets of queries to those launched by ODoH were also launched over 8 additional DNS variants, summarized by Table 6.2. UDP DNS on port 53 (Do53) and DNS over HTTP (DoH) form our 'best-possible' baseline, in which clients direct queries randomly to one of the three resolvers.

To better characterize different aspects of the ODoH path, we implemented two artificial proxied variants that inject ODoH features to DoH in a stepwise fashion. First, in 'proxied DoH' (pDoH) the client requests proxies to forward Do53 queries as DoH queries to a chosen resolver; this is an architectural variant of the DoH protocol that introduces one additional hop between the client and recursive resolvers. The additional hop is then converted to DoH in 'cleartext ODoH', a two-hop DoH query without end-to-end encryption. Alongside, we evaluate ODoH when the target and resolver are co-located, the model that we anticipate will be dominant in practice.

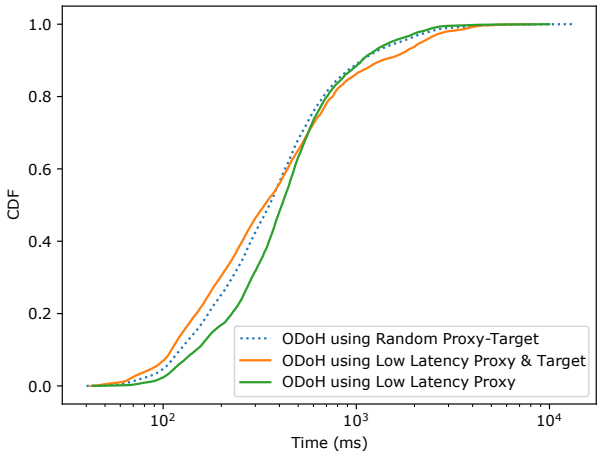
The DoH and ODoH variants described above allow us to compare conventional DNS metrics,

such as performance, but fail to capture properties related to privacy. For this reason we also mirror the same query measurements with DNSCrypt, which encrypts UDP queries between client and resolver. Both privacy-oriented Anyonymous DNSCrypt and DoH over the Tor network (DoHoT) introduce proxies to achieve privacy. We used dnscrypt-proxy [105, 132], an open source implementation of the DNSCrypt proxy, and restrict its use to only the DNSCrypt protocol; TCP was disabled, according to the original specification,and dnscrypt-proxy was configured to generate a new ephemeral key for each query for fair comparison against ODoH. Our stub resolver was configured to the lowest-latency DNSCrypt resolver from the public list of DNSCrypt resolvers available. DoH over Tor (DoHoT) was granted similar performance benefits. Specifically, we allowed Tor to create its own optimal circuit, then performed DoH queries over Tor using the SOCKS5 proxy on each client instead of configuring them explicitly to use a static route. However, a limitation of our measurements with Tor are that the set of optimally chosen nodes might not all reside within the United States matching the comparisons to other protocols.

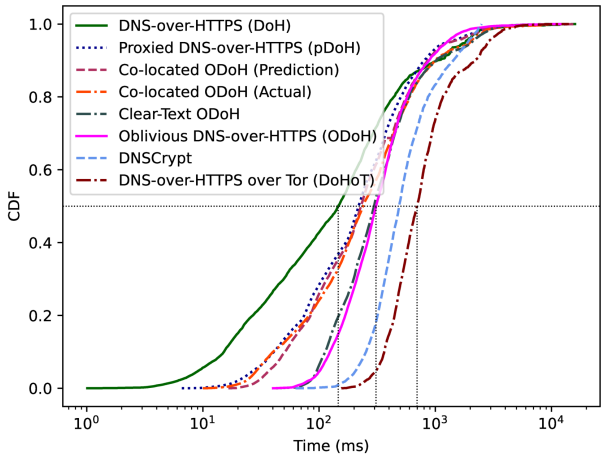
6.1.5 Results

We first look at various ODoH query-path selection strategies, and then compare the response latencies with other DNS protocols and network architectural variants summarized in Table 6.2.

Proxy and Target Selection: We configure the client stub resolver to select proxies and targets in three ways: (i) random selection of a proxy and a target pair; (ii) lowest-latency proxy with random target; and (iii) the lowest-latency proxy-target pair, i.e. fastest ODoH path to any target. The measured response times are shown in Figure 6.2a, and indicate that the choice of proxy and target does impact ODoH performance. These measurements were performed with the separation of the target and resolver. Observations from Figure 6.2a suggest the best overall selection strategy is lowest latency proxy-target pair. Intuitively this choice makes sense, and is most pronounced when the proxy is near- or on-path to the target. In these evaluations the median query to response



(a) ODoH query response times with different proxy and target selection strategies. Measurements suggest lowest total latency to target as the best strategy. The high performance of random selection is likely due to clusters of clients, proxies, and targets along eastern and western regions of North America.



(b) Comparison of ODoH to other DNS Protocols. ODoH performance matches proxied DoH (pDoH) when the target and resolver are co-located. Note that both colocated ODoH and non-located ODoH variants provide privacy at a substantially reduced cost over other privacy-oriented protocols. (Left to Right order of curves maintained in legend order)

Figure 6.2: Comparing path choices and latency impacts of ODoH to other DNS protocols

time over the lowest latency proxy-target pair is 334.85 ms compared to 411.44 ms when solely selecting the lowest latency proxy. This marks a median improvement of 22.8%.

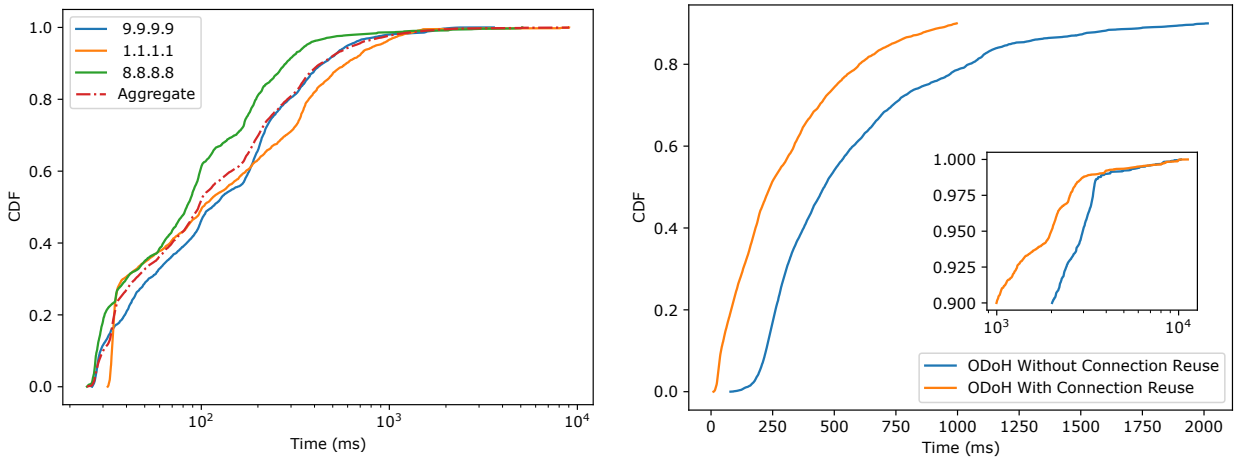
We also observe that random selection is high-performing than averaging across all queries; we suspect this is because most clients in our deployment have a higher than average set of nearby proxies and targets from which to choose, i.e. clusters along the western- and eastern-halves of North America.

Comparing DNS Protocols: Figure 6.2 shows the query performance of ODoH instantiations compared to other DNS platforms that are summarized in Table 6.2. ODoH is the best performing privacy enhancing DNS protocol, ahead of DNSCrypt and DoH over Tor. Also, ODoH performance is only marginally worse than ODoH in cleartext, where a standard DoH query traverses the

same logical 3-hop path as an ODoH query. This reinforces the minimal cryptographic compute overheads introduced. Observations from Figure 6.2 also reinforce that the proxy and target selection dominate performance. This can be seen by comparing standard DoH as the baseline to pDoH and ODoH variants. The median DoH response of 146.12ms increases by 49.5% to 218.52ms when the same DoH query is proxied (labeled as pDoH with the dotted black line). The same DoH queries increases the median response time by 109% when using the ODoH protocol to 305.11ms and in the case of Cleartext ODoH, when sent as DoH through both the proxy and the target, before being resolved by the resolver, we see an increase in median DNS response times by 103.2% to 297.015ms. We estimate, however, that the additional penalty of ODoH over pDoH is mitigated by co-locating the oblivious target with the resolver.

If we focus on privacy-performance trade-offs, the ability of ODoH to match proxied DoH variants is particularly noteworthy: Compared to a median ~50% increase in co-located ODoH queries, the median DNSCrypt query time of 487.68ms is 233.7% greater than DoH, while DoHoT is 376.5% greater at 696.40ms.

Colocating Targets and Resolvers: The specification defines Oblivious Targets as being independent from resolvers, but allows for their co-location or integration. While we envision direct ODoH support in popular DoH providers, in practice, we make the same distinction as the specification in our evaluations – measurements labeled as ODoH consist of targets that are physically separate and hosted individually, while *co-located ODoH* targets are implemented within resolvers' datacenter facilities or in the resolver services directly. Crucially, either instantiation preserves all previously stated security and privacy properties. The effects of target location on performance are hinted at by closer inspection of the time it takes to resolve queries from any single target. For example, Figure 6.3a shows query resolution times from our target deployed inside Google Cloud datacenters. From this target, queries to Google's DNS service consistently resolve in less time than queries sent to resolvers not located within Google's datacenters.

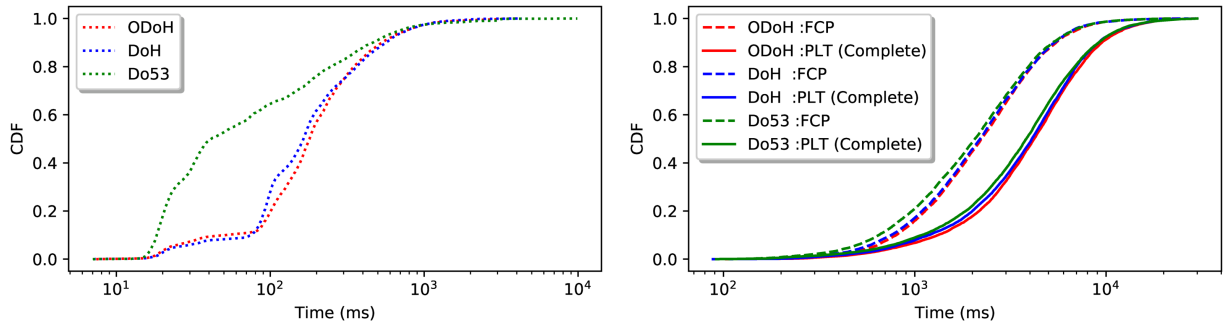


(a) The time taken to resolve DoH query from target instances to resolvers points to colocation benefits. Here, almost all queries from the target deployed in Google Cloud are fastest when directed to Google's own DNS resolver.

(b) ODoH Connection reuse: ODoH query resolution times increase if a new connections is used for each query.

Figure 6.3: Impact of Target Co-location and Connection Reuse on ODoH

We use this insight to estimate the values for co-located ODoH service in Figure 6.2 (indicated by the dashed blue line). Doing so allows us to isolate query resolution time at the resolver from the cost of the last target-to-resolver hop. We estimate the improvement by measuring a 'toy' recursive DNS resolver, in which queries resolved by a cache hit take less than 1ms at the 99^{th} percentile, and take 120ms with a cache miss. We conservatively assume a cache hit ratio of 70%, and use this value to estimate the lower-bound median response times of 228.52ms if Oblivious Targets were always co-located with the resolver. We further validate these estimates with live measurements. The ODoH services were hardened for scale and implemented into a large public recursive resolver operated by Cloudflare, enabling us to measure ODoH with production targets. Both the predicted and measured values (purple dashed and dot line) are indicated in Figure 6.2. Their performance is similar to proxied DoH, with the median response time from co-located ODoH being 238.60ms, suggesting that predicted median value 228.52ms are reasonable, as are



(a) DNS response times for 10000 Page Loads from Local Stub Resolver

(b) First Contentful Paint (FCP) and Page Load Time (PLT) measurements for 10000 webpages based on DNS protocol used

Figure 6.4: Impact of page load time due to usage of ODoH compared to other DNS protocols

the potential performance improvements of co-location.

Connection Reuse: Persistent connections, or connection reuse, are HTTPS optimizations that enable clients to establish and re-use a connection to transmit multiple requests to the same host, rather than to open a new connection for each request. In all of our experiments described from here, the proxy-target connections are persistent and use the co-located target case. However, the measurements presented in Figures 6.2a and 6.2 benefit from persistent client-proxy connections for best performance. The use of persistent connections does not limit the ODoH security properties; however, we consider the impact of initiating a new HTTPS connection for each query in Figure 6.3b. In our measurements the median query response times with a client creating a new connection for each request without reuse, increase by 92.5% from 238.60ms to 459.42ms – indicating the value and the need for connection reuse.

Page load times with ODoH: Previous DoH evaluations concluded that the performance penalty of DNS-over-HTTPS over clear text DNS is absorbed into page load times with no significant perceived effect [60]. We similarly evaluate page load times with ODoH by replacing the

DoH stub resolver in a popular open source DNS proxy with an ODoH compatible stub resolver. In this experiment a set of 10000 websites selected, composed of the top 5000 from the top million Tranco dataset [267], and another 5000 randomly selected from the remainder of the list. Page load times are captured across three configurations for both the top 5000 and the randomly chosen 5000 websites: (1) Do53 to Cloudflare's 1.1.1.1; (2) DoH to 1.1.1.1; (3) ODoH with an off-path proxy to a colocated ODoH resolver; The client stub resolver reuses connections when possible.

To measure page load times we use Google Chrome and capture the page load session into an HTTP Archive (HAR) file additionally recording the results of the window.performance API [253]. Each experimental run is launched with a new profile to remove caching effects and uses the same set of 10000 websites loaded in the same order. This is done to ensure that the results captured during the page load time measurement does not include order effects. We believe that loading the same set of pages, from the same starting state, in the same sequence would balance out additional effects due to browser artifacts such as caching. Maintaining the sequence of page loads across all the experimental settings indicates the same cache state of the browser after each web page load with a high probability. Additionally, the local DNS stub resolver cache is flushed for each configuration forcing an empty starting state for each evaluation. Page load time is calculated from page load event information in the defined HAR format, and measure the DNS lookup times as the difference between domainLookupEnd and domainLookupStart events recorded by the browser. These values are validated with the DNS time for the request captured in the HAR file. This is recommended by the W3C standard for page navigation time and perceived page load time [346].

We present our results on page load times with different DNS protocols in Figure 6.4. Figure 6.4a shows the time taken to resolve queries for the websites. The median DNS response time when using Do53 is 41.28ms and increases by 298.01% to 164.30ms for DoH and 335.70% to 179.86ms for ODoH indicating a 9.47% difference between usage of DoH and ODoH. The CDF results for DoH

and ODoH indicate a sharp increase close to 100ms possibly indicating an artifact of the local stub resolver. Page load measurements for the websites are shown in Figure 6.4b. The median page load time when using Cloudflare DNS in cleartext UDP Do53 packets is 3830.22ms. We consider this the baseline and find that the median page load times increase by 6.76% when using the DoH protocol to Cloudflare resulting in 4089.35ms median page load time respectively. The usage of colocated ODoH with the resolver increases median page load times by 9.83% compared to the cleartext baseline usage to 4207.10ms. To understand the impact of colocation in page load performance, the usage of ODoH with separation between the target and the resolver increases the median page load time by 20.12% to 4601.03ms. At the 95th percentile, we notice that the difference between DoH and ODoH protocols compared to the baseline cleartext UDP based DNS protocol reduces to 0.66% and 5.86% respectively resulting in page load times of 11524.59ms and 12113.84ms when compared to 11448.89ms with Do53 Cloudflare DNS.

The performance difference between ODoH and DoH (and Do53) can be attributed to the network path topology differences. With an on-path proxy to the resolver, clients can benefit from slightly better page load times for the ODoH protocol. The complete page load time measurements might skew results due to intensive client side javascript operations. To mitigate, in Figure 6.4b we also present our results for the First Contentful Paint (FCP) measurements for the website loads and identify that the median FCP time for client using the Do53 protocol is 2065.17ms and increased with the usage of DoH protocol by 6.15% to 2192.33, and ODoH by 8.03% to 2231.12ms which are in-line with the total page load time measurements indicating the impact of protocol choice.

Our results indicate that client choice of proxy and targets strongly impacts the user-facing performance of the ODoH protocol and the corresponding page load time measurements. Additionally, we show that co-location of the oblivious target services with the recursive resolver improves performance. Our experiments show that the median page load times increase by 2%

when using ODoH compared to secure DNS protocols like DoH while providing both security and privacy guarantees making it a practical replacement for the Do53 DNS protocol and could be used by clients adopting DoH needing additional privacy guarantees. We further believe that the usage of ODoH will have no adverse impact on video streams or gaming since there is no constant need to rely on DNS once an IP address is obtained and a successful connection is established.

6.1.6 Potential Attacks & Implications

Beyond the boundaries of ODoH privacy properties, attacks may be possible. These attacks are either prohibitively expensive, or are due to known vulnerabilities of the underlying mechanisms.

Association attacks: A requirement of ODoH is that per-query proxy and target pairs are selected from non-colluding parties. There remains, however, a few scenarios in which it becomes possible to link or infer queries with clients' subsequent request activity. For example, a selected target within the clients' ISP could conceivably be used by the ISP to link IP addresses returned by the target with subsequent requests to those addresses. An Oblivious Proxy selected within the ISP could be used in a similar manner. Specifically, an ISP might be able to deduce the contents of a query, and link that query to a client, if the subsequent request or its destination IP address reveal any information about the activity. The usage of ODoH in these scenarios increases the effort needed to reliably track clients and is no more or a risk than what already exists with DoH.

Denial of service (DoS): Resolvers may throttle IP addresses associated with malicious clients attempting to perform denial of service (DoS) attacks. ODoH would seem to shift this responsibility downstream to the Oblivious Proxy or Target that is receiving the malicious queries. Proxies should therefore implement rate limiting or other mitigation strategies like the usage of an allow list of targets to prevent clients from launching large numbers of DNS requests and choosing arbitrary targets in an attempt to perform DNS Tunneling attacks or as a form of abuse. Malicious

clients can however distribute their requests to different proxies destined to the same targets. Targets should implement similar strategies to protect against malicious proxies, and contact proxy operators. Well-behaved clients that are negatively impacted by target rate-limiting of the selected proxy could shift queries to different proxies. Alternatively, malicious clients could also encrypt malformed DNS queries or check for non-existent domains in an effort to make the resolver spend compute cycles in decryption of the message or spend time communicating with various name servers [63]. We argue that with reasonable DoS prevention in place for deployed artifacts, such amplification or denial of service attacks can be made impractical in practice.

Public Key linkability: ODoH targets may give each client a unique HPKE key for query encryption in an attempt to deanonymize them. This is a common problem with protocols of this form, including Privacy Pass [101]. Multiple mitigations exist against this type of attack. For example, a trusted third party may fetch target keys on behalf of multiple clients, so that a target is unable to uniquely tag clients with public keys. The exact mechanism for mitigating against this problem is a general problem that is far from unique to ODoH.

Compromised proxies and targets: An adversary that wants to link queries to clients must compromise both proxy and target. Given the proxy's (IP, C_Q) tuples and the target's (C_Q, Q) tuples, the attacker in possession can perform a JOIN to map client IP addresses to their queries. In a practical setting, a malicious proxy can correlate encrypted queries to encrypted responses (C_Q, C_A) but does not gain much due to replay attack resistance. Network adversaries at the ingress and egress could perform correlation attacks but are practically limited due to TLS [14].

Encrypted DNS protocols pose challenges to enterprise system administrators that rely on DNS to keep organizations safe or provide controls that protect users inside the network from content deemed harmful or malicious. In these situations, we believe ODoH can be used to increase organizational safety. For example, an enterprise could use DoH for queries to the internal resolver,

and use ODoH for any outbound queries from the organization's gateway. This protects potentially sensitive organization web traffic information from being recognized by upstream ISPs and other DNS name servers. Techniques to protect Internet traffic while maintaining local control over DNS can also be implemented in a home network with tools such as PiHole [157]. For example, the Internet gateway for all web traffic from the home can convert insecure Do53 queries to ODoH.

6.1.7 Conclusion

Cleartext DNS queries continue to be the most popular way in which users communicate with DNS ecosystem. By implementing, analyzing, and evaluating Oblivious DNS over HTTP and comparing it to its secure and private counterparts we conclude that ODoH guarantees that only the client has knowledge of both its query and its IP address. Decoupling the trust between the proxy and the recursive resolver provides a practical and highly performant mechanism to address the challenges with user privacy. It allows Internet users to opportunistically upgrade their protocol usage and reduce the implicit trust that's placed in the correctness of operations and validity of the answers provided by their DNS resolver. Our evaluations show that this level of privacy incurs marginal performance cost relative to DoH, while substantially out-performing privacy-oriented DNSCrypt and DoH over Tor. Overall, we find that ODoH has the potential to be deployed at scale with good performance, while also contributing to a safe and secure Internet.

6.2 Leveraging Internet Co-location for Privacy

Consolidation of infrastructure poses various risks and raises challenges similar to the current state of cellular networks outlined in §4 – ranging from infrastructure resilience challenges due to the single point of failures, loss of privacy and control over personal data within a few select organizations. On the flip side however, the consolidation of infrastructure among a select few organizations, typically the large public cloud and CDN network providers has enabled scalability, improved infrastructure flexibility and reduced capital expenses for organizations – and arguably as played a significant role in improving the state of Internet security by accelerating TLS adoption [128]. The increasing co-location of content and services on the Internet open up interesting opportunities to deliver more private and performant experiences to the users. We perform large scale Internet measurements to understand the missed opportunities – reusing connections and preventing expensive DNS lookups (especially due to improved privacy enhancing protocols), the extent to which resources are co-located on the Internet [313].

6.2.1 How Co-located are resources on the Internet today?

The HTTP protocol and specifications evolve in response to needs and insights generated by web applications [50, 251]. Slower to evolve are the conventions with which web applications are developed and deployed. Understanding the gaps can help to identify best practices, generate consensus, and shed light on shortcomings. One recent advancement is *connection coalescing*, a feature enabled by the HTTP/2 specification that enables connection re-use for different hostnames and domains, so long as the server is authorized to serve them [251]. The benefits of connection coalescing are best understood with the context of HTTP, which began as a one request, one connection, protocol [52]. Pipelined requests and persistent connections were later introduced in HTTP/1.1 [129] to improve performance [41]. The gains were themselves bottlenecked by head-of-line blocking [193, 248].

This led to a practice of domain sharding, in which browsers are 'tricked' into initiating new and parallel connections to multiple subdomains [141]. As a result, the burden of managing multiple connections, scheduling requests, and optimizing rendering pipelines shifted to browsers. As the web continued to grow and evolve, web page load times (PLT) were understood to be a key performance metric, affecting user experiences and business revenues significantly [121, 327, 345]. To address continued performance limitations of HTTP/1.1, SPDY, a completely backward-compatible extension to HTTP/1.1, was introduced [344]. SPDY is a binary protocol, in contrast to text-based HTTP/1.1, and supports stream multiplexing where requests to and responses from the server use a single interleaved TCP connection [344]. SPDY also introduced prioritization of requests allowing browsers to request critical resources blocking page rendering first. A modified variant of SPDY was standardized by the IETF as HTTP/2 in 2012 and published as RFC 7540 in 2015 [50]. Despite the ability with HTTP/2 to revert sharding, little about website maintenance practices have changed. Alongside, HTTP/2's ability to coalesce websites that rely on otherwise unrelated domains is further complicated by certificate issuance, management, and maintenance to keep pace with websites and changes.

ORIGIN Frames were later proposed by CDN operators, and introduced into HTTP/2, so that content operators could signal coalescable connections in a way that reduces the certificate management burdens and facilitates scale and operational flexibility [251], while leaving the verification and correctness to the clients. To the best of our knowledge, however, there exists no server-side implementation of ORIGIN. Even without ORIGIN, studies show that opportunities to coalesce based on IP address are being ignored or overlooked. Instead, browsers use redundant and excessive numbers of connections to request content because of longer-standing domain sharding and DNS load-balancing conventions [291], speculative optimizations [333], and varying implementations of Same-Origin Policy implementations [305].

We used 100 2vCPU 7.5GB memory virtual machines in Google Cloud's East US datacenter,

Tranco sites		Per-website Median Values			
Rank	**Success**	**#Reqs**	**PLT (ms)**	**#DNS**	**#TLS**
1-100K	68244	89	6168.0	17	20
100K-200K	64163	83	5720.0	14	17
200K-300K	63334	80	5601.0	14	16
300K-400K	59827	79	5565.0	13	15
400K-500K	60228	78	5724.0	13	15
Total	**315796**	81	5746.0	14	16
		μ = 113	μ = 8088.0	μ = 22.55	μ = 26.59

Table 6.3: Successful automated collection of top-ranked Tranco websites, with median page-level attributes. Failures caused by non-200 HTTP errors and CAPTCHAs appear to distribute evenly across the set.

each running a modified privately hosted version of Web Page Test (WPT) and requested the top half million webpages from the Tranco dataset [205, 225]. Data was collected between 14 to 18 February 2021 with no network traffic shaping, allowing browsers to use available bandwidth freely. A designated primary VM instance was responsible for queuing list entries and instructing replicas to execute the data collection. For each root web page in the queue, WPT instances initiated a new Google Chrome (v88.0) session to eliminate DNS, and resource caching effects. On successful completion, Chrome developer tools were used to retrieve and write the page load data as an HTTP Archive format (HAR) file with a full timeline of events, as well as additional information about certificates and their validation. Resulting HAR files were stored into a cloud storage bucket.

From the 500K websites that the WPT instances attempted to retrieve, 315,397 (63.51%) succeeded. The remaining trials were removed from the dataset because they either received non-200 response codes or were met with CAPTCHA challenges. Our snapshot of 315796 webpages results in 35,882,587 total network requests for completing the page loads. A coarse view of the trials is summarized in Table 6.3, binned by popularity rank into 100K size buckets. Interestingly, median values across the set appear to be unaffected by popularity rank. Across the whole dataset,

webpages initiate requests for a median 81 subrequests (μ: 113 subrequests, IQR: 90). These additional HTTP requests incur medians of 14 additional DNS queries and 16 TLS handshakes and verifications, which contribute to a median 5746.0ms page load time for rendering the full page.

Finally, the destination IPs for each request were resolved to its origin autonomous system (AS) so that we could identify the number of requests serviced by any AS for each page. Additionally, we parsed the certificate chains of the TLS for the root webpage and new TLS connections triggered by subresource requests to (i) identify certificate issuers for the hostnames, (ii) inspect the presence of the Subject Alternative Name (SAN) extension, and (iii) validate that DNS names resolve to the IP address used. Each request was also parsed to determine the version of HTTP used.

Sources of selection bias

Our dataset consists of a single snapshot. A longitudinal study to capture code churn, sub-resource modifications, and other website changes is out of scope. All data is collected by machines in a single datacenter, which might experience race-conditions or network effects that could be elucidated by geographically distributed scanners. We used the same machines throughout our evaluations for consistency, and saw no observable change in datacenter provisioning or capacity over time. However, our single datacenter source of data collection on 500K websites could have triggered some of the DDoS protection measures and CAPTCHAs, which was mitigated by pacing the rate of website requests. The values in Table 6.3 give an indication that any possible bias or skew is unaffected by popularity.

Page load characterization

In our dataset, requests were made to a total of 13316 ASes. The top-10 destination ASes for requests to *sub-resources* are listed in Table 6.4, and belong to eight unique providers. The top-3 providers (Google, Cloudflare, and Amazon), represented by 4 ASes, service ~50% of all requests

Rank	AS Number	Org. Name	#Req	%
1	AS 15169	Google	7932198	22.10
2	AS 13335	Cloudflare	4937395	13.75
3	AS 16509	Amazon 02	3017176	8.40
4	AS 14618	Amazon AES	2019308	5.62
5	AS 54113	Fastly	1281402	3.57
6	AS 16625	Akamai AS	1087172	3.02
7	AS 32934	Facebook	998685	2.78
8	AS 20940	Akamai Intl. B.V.	583700	1.62
9	AS 16276	OVH SAS	548107	1.52
10	AS 24940	Hetzner Online GmbH	469293	1.30
Total				**63.68**

Table 6.4: The top-10 destination ASes for resource requests in our dataset. Two providers in the list each operate two ASes in the top-10.

in the dataset. Collectively, the top-10 ASes (0.06%) service more than 60% of the total requests in our dataset. Looking beyond the table entries, it takes 51 ASes (0.38% of ASes in the dataset) to service 80% of the requests.

The number of unique ASes needed to fully load a webpage is summarized by Figure 6.5. The proportion of webpages that rely on a given number of ASes (blue-line, y_1-axis) shows that 6.5% of webpages request subresources from a single AS; these webpages could immediately benefit from connection re-use. The largest bin indicates 14% of webpages need two ASes to fully load i.e. pages that have a dependency on one additional AS for subresources. The corresponding CDF (red-line, y_2-axis) shows that 6 ASes are needed to fully load more than 50% of all webpages, suggesting very high colocation of content with the necessary subresources. Notably, from this perspective, the potential for connection re-use is optimistically approximated by the number of unique ASes needed to retrieve all subresources in a webpage. In practice, coalescing opportunities may be superceded by operational factors such as commitments to service level agreements, or helped by flexible mappings between sockets, names, and IP addresses [126].

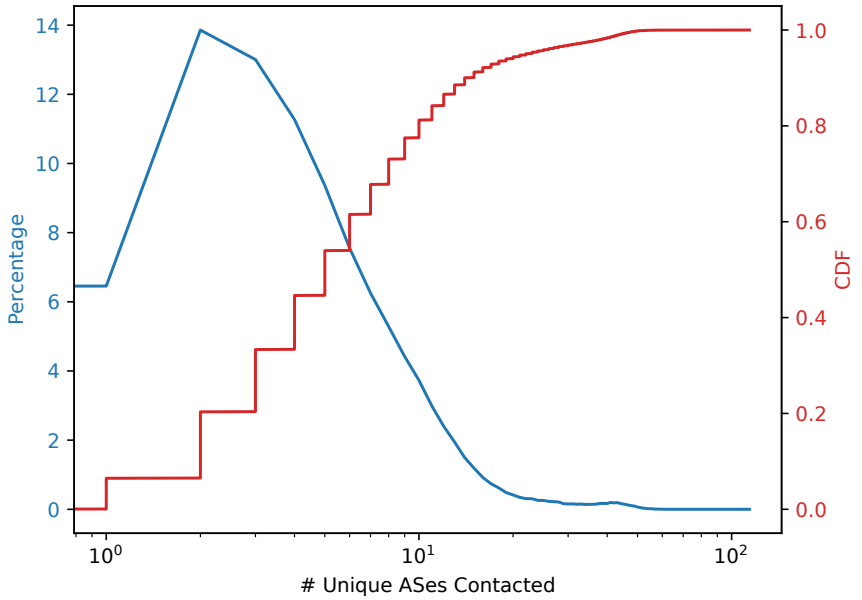


Figure 6.5: The frequency distribution (y_1-axis) and CDF (y_2-axis) of the number of unique ASes needed to load a webpage in the dataset

By analyzing the connection level information, we observe that 98.53% of the requests (35356995) are HTTPS connections, which validate server authority for the requested resources. The remaining 1.47% using HTTP can still re-use connections but require IP address matching via DNS. The majority of requests use HTTP/2 at 73.64%, with 19.09% of requests using HTTP/1.1 as negotiated by the clients. The 0.41% proportion of HTTP/3 and QUIC requests outnumber HTTP/1.0 and HTTP/0.9. Among the total requests made during our data collection, we find that 16.24% (5827389) initiated new TLS handshakes and validated corresponding certificates. The top-10 issuers of certificates for sub-resources are shown in Table 6.5. The top-5 distinct certificate issuers cover 59.25% of all certificates issued; if we count by the top-5 providers (aggregating multiple DigiCert services) this proportion grows to 69.05%. 3 of the 8 certificate issuers in Table 6.5 also can host content or terminate connections on origins' behalf. Providers of both services *may* be able to modify certificates to enable coalescing.

Certificate Issuer	**# Validations**	**%**
Google Trust Services CA 101	1507140	25.86
Let's Encrypt (R3)	558798	9.58
Amazon	533391	9.15
Cloudflare Inc ECC CA-3	443727	7.61
DigiCert SHA2 High Assurance Server CA	411107	7.05
DigiCert SHA2 Secure Server CA	405377	6.95
Sectigo RSA DV Secure Server CA	402757	6.91
GoDaddy Secure Certificate Authority - G2	181798	3.11
DigiCert TLS RSA SHA256 2020 CA1	166198	2.85
GeoTrust RSA CA 2018	93168	1.59
Total (16.24% total requests)	**5827389**	**100.0**

Table 6.5: Breakdown of top certificate issuers by number of cerificates validated in the requests

In practice this ability is limited by certificate management complexities, or when customers choose to use separate or multiple providers. Similarly, analyzing the requests organized and ranked by sub-resource content types is presented in Tables 6.6a and 6.6b respectively, in addition to grouping the content type by top ASes servicing those subresource requests. In both rankings across the dataset and for the top-3 ASes, Javascript is the most requested subresource content type. The top-10 subresource content types in the dataset are responsible for 86.55% of all subresource requests. When organized by provider, the most requested subresource types are similar. However, Google is unique for servicing a high number of requests for HTML and fonts resources, and for listing javascript as a $\mathrm{text/javascript}$ media type, which is now obsolete [155].

We observe that surprisingly, the top 10 hostnames which are used to request subresources in the top half a million web sites on the Internet account for 12.5% of the total requests. This presents a view into the consolidation of the critical Internet resources today and suggests opportunities that we may have been missing all throughout.

Content Type	# Req	%
application/javascript	5118428	14.26
image/jpeg	4674423	13.02
image/png	3831287	10.67
text/html	3703184	10.32
image/gif	3221105	8.97
text/css	2796536	7.79
text/javascript	2428665	6.76
application/json	1270236	3.53
application/x-javascript	1208062	3.36
font/woff2	963218	2.68
image/webp	960160	2.67
text/plain	904010	2.52

(a) Requests breakdown by Top 12 content types

ASN	Content Type	#Req	%
Google (AS 15169)	text/javascript	1720366	21.69
	text/html	1141738	14.39
	image/gif	869759	10.96
	font/woff2	792852	9.99
Cloudflare (AS 13335)	application/javascript	1102387	22.32
	image/jpeg	959775	19.43
	image/png	590532	11.96
	text/css	529758	10.72
Amazon 02 (AS 16509)	application/javascript	644643	21.36
	image/jpeg	442718	14.67
	image/png	405607	13.44
	text/css	205720	6.81

(b) Top 4 content type requested from Top 3 ASes

Table 6.6: Breakdown of requested content type and their top serving autonomous systems

6.2.2 Finding the Extent of Missed Opportunities on Today's Internet

To understand connection coalescing as it exists and as it could be, we first extracted the webpage request timelines from each HAR file in our dataset. From these, the individual subresource requests were inspected and analyzed to identify those that could have been coalesced if the corresponding hostnames had been included in stream 0 as an ORIGIN Frame. Each timeline was then reconstructed first by finding the timelines' event labels {*block*, *send*, *wait*, *receive*} for the affected subrequests. We then modified those timestamps, conservatively, by omitting the smallest DNS query and TCP/TLS connection establishment times for blocking requests.

Consider a waterfall representation where multiple coalescable requests start at the same time but have varying DNS response times. The smallest time to remove is the minimum of the DNS response times for these queries and the difference between the larger response times is retained in the new timeline reconstruction. The CPU time beforehand to decide and dispatch the subresource queries and connections is unmodified in an effort to model browsers' dependency

graph computation time. The resulting request timelines represent page loads on the basis of coalescing that could have been possible given ideal certificate SAN, ORIGIN Frames from operators, and client support.

Informed by earlier observations (§6.2.1), we assume that every server in each ASN can authoritatively serve all the content for that ASN. We believe this to be reasonable for smaller operators that are likely to control larger proportions of their content (eg. universities). For larger operators, however optimistic this assumption, prior works suggest that this approach is feasible [126]. Recent measurements have shown that it is possible to reach around 50% of all accessible webpage in the .com, .org, .net, and *.info* in less than 100 IP addresses and over 80% with upto 10000 unique IP addresses. This massive re-use of hostnames indicates that such practices are possibly in effect already [322]. The decision to equate an ASN with its ability to coalesce connections forms the core of our model and identifies both operational opportunities, as well as a potential limitation of the model's projections.

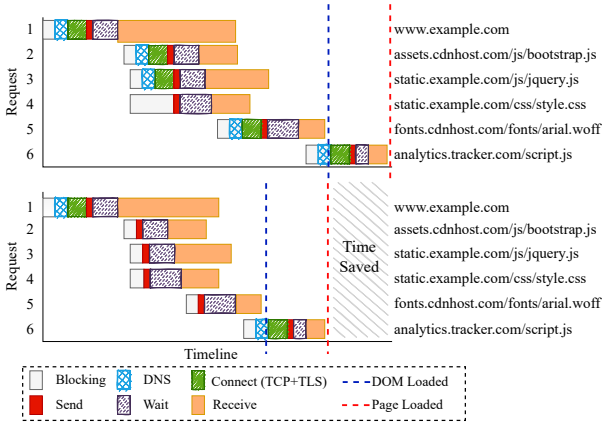


Figure 6.6: Timeline of requests involved in a page load event (top) with a reconstructed timeline due to ORIGIN Frames (bottom) showing an improvement in DOM loading and page load time.

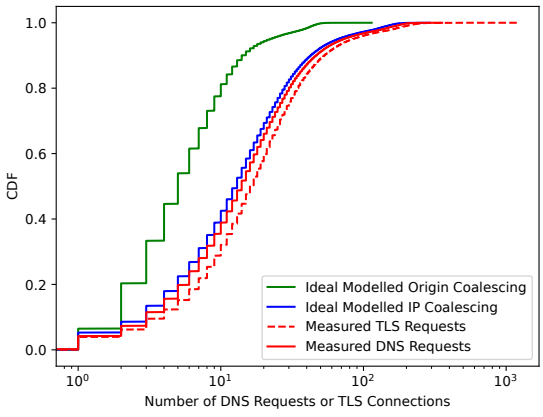


Figure 6.7: Comparison of the measured number of DNS requests and TLS connections using Chrome to the ideal IP based coalescing and ORIGIN based coalescing settings

An example reconstruction is depicted in Figure 6.6 in which 6 object requests are needed to load www.example.com served by the CDN cdnhost.com. The top timeline draws from the original HAR file, and beneath it is the reconstruction if the requests were coalesced. As the browser parses and validates the base HTML, requests 2, 3, and 4 are initiated for critical subresources. The css resource retrieved by request 4 prompts the browser to initiate request 5 for a font, and finally leads to a *low priority* request 6. The request for a base-page can never be coalesced since it initiates the first connection. In the example in Figure 6.6, request 2 to assets.cdnhost.com, given an ORIGIN Frame, could be coalesced onto the existing connection because the resource is available on the same CDN that serves the base-page. Similarly, each of requests 3, 4, and 5, are to sharded domains that are reachable on existing connections and can be coalesced. The last request (6) attempts to fetch resources from a hostname which is not serviced by the same CDN and hence cannot be coalesced. For purpose of modelling and prediction, we discard *DNS* and *Connect* events for coalescable requests, then reconstruct the timeline, accordingly. Before deciding that sharded domains can be coalesced, we first inspect IP address sets returned by DNS. For sets that reveal transitivity, we then identify the origin AS of the IP addresses. The IP to ASN mappings are drawn from an internal database at Cloudflare.

The set of names that should appear in an ORIGIN Frame for a website are those that could have been coalesced. The timeline reconstructions, coupled with the characterizations of Internet co-location, give us a model to predict the impacts of ORIGIN Frame-based coalescing. Notably, this example makes clear that the *schedule* of events to load a page is compacted as due to coalescing.

6.2.3 Predicting Queries, Connections and Certificate Validations

In an *ideal* coalescing, the number of DNS queries, TLS handshakes, and certificate validations is equal to the number of separate services (not domains or hostnames) needed to serve all webpage resources. The ideal requires that, and for the purpose of prediction, we assume that

ORIGIN Frames and certificate SANs are correctly configured with each service sending a valid corresponding ORIGIN Frame when a new connection is initiated.

DNS queries and TLS connections

The measured and ideal number of DNS queries and TLS connections needed to load a webpage are summarized by their CDFs in Figure 6.7. Interestingly, in our dataset the measured distributions differ between DNS queries and TLS connections, with medians of 14 and 16, respectively. Closer inspection suggests the differences are a result of race conditions in the browser. Examples include, (i) the practice of "happy eyeballs" [298] in which clients simultaneously initiate connections over both IPv4 and IPv6 to mitigate the performance penalty for failure on either; (ii) also speculative behaviour [169] which causes sets of sub-resources to be fetched in parallel by multiple network-state management threads (eg. fonts from a CSS file). These types of race conditions generate additional DNS queries or make multiple connections for the same sets of resources.

Alongside measured values in Figure 6.7 we plot distributions under ideal coalescing conditions. For IP-based coalescing, we inspect each webpage load in the dataset and reduce any set of two or more connections to the same IP address, to one connection; in other words, our model assumes no changes and looks for 'missed opportunities'. The median number of DNS queries and TLS connections for IP coalescing is 13, only ~7% less than measured DNS but ~19% for TLS and is presented as the solid blue line in Figure 6.7. These values are the upper-bound in which no two hostnames are listed on a single certificate. Certificate changes are operationally complex. In contrast, the ideal coalescing with ORIGIN Frames indicated by the solid green line in Figure 6.7 show marked improvement with a median of 5 each of DNS queries and TLS requests. In our dataset, this reduces median DNS queries by ~64% and TLS connections by ~67%.

Certificate Validations

As ORIGIN-based coalescing reduces the number of TLS connections, it also reduces the number of cryptographic certificate validations. As a result, the ideal ORIGIN coalescing for our dataset in Figure 6.7 predicts a best-case median of five certificate validations. For this distribution the interquartile range reduces from 22 in the dataset to 6; the 75th percentile shows a 76.67% improvement reducing the number of certificate validations from 30 to 9. One implication on the client is that the cryptographic computation overhead for validations is substantially reduced.

A properly supported ORIGIN Frame reduces the number of TCP and TLS connections as predicted by our model. However, full reduction of DNS queries and certificate validations presented in our model requires that the names in the ORIGIN are also listed in the certificate SANs. In the absence of the hostnames in the certificate, the client must query DNS and, if the IP address matches the address of the current connection could, *at best*, reuse the already established TCP connection to initiate a new TLS connection. In practice, the reuse of an existing TCP+TLS connection for a new TLS connection requires termination of the existing TLS connection. Instead of keeping the existing TCP connection, browsers create new TCP+TLS connections. ORIGIN Frames reduce this burden on clients. By adding SANs to certificates and coalescing, fewer TCP and TLS connections need to be established. This raises a series of questions about the certificate changes needed in support.

Generate new certificates or modify existing ones?

Certificate reuse is expected and common practice, but the scale and quantity of certificates to modify or generate has operational constraints: A single large certificate with all hosted names (as many as millions [126]) is unreasonable; equally, having as many certificates as webpages is challenging for operators to manage effectively. A full measurement and understanding is

regarded as out of scope: The scale (Let's Encrypt issues over 2.5 million certificates daily [2]), as well as individual operators' constraints, deserves its own investigation and is left for future work.

Our model takes a compromise position and assumes no change in the number of certificates. Instead, we determine the volume of changes to existing certificates needed to enable perfect coalescing. To understand those changes we identify the validated certificates for each website in our dataset. In each website's certificate we identify and add the individual hostnames needed to load the webpage that are available from the same provider but absent from the SAN.

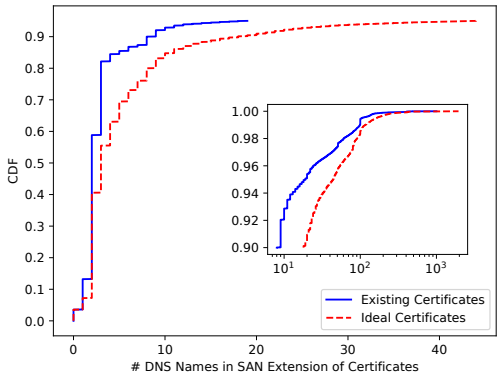


Figure 6.8: Comparison of the number of DNS SAN names in Existing Certificates (blue) to those after proposed certificate modifications (red). The top-most outset y-axis value is 0.94.

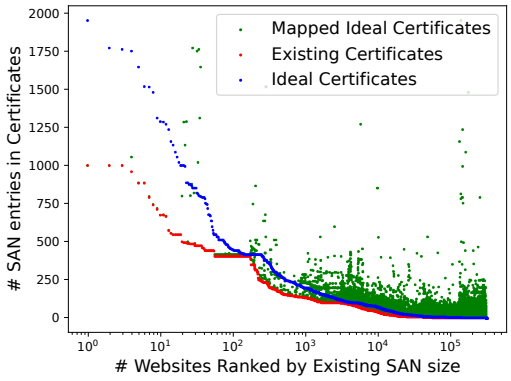


Figure 6.9: Tail distribution of number of SAN entries in existing certificates (red) with those after their modification (blue) for all navigated websites in the study. The changes needed per certificate in the existing distribution are shown as mapped ideal certificates (green).

Figure 6.8 shows the distributions of SAN entries of the existing set (solid blue line), and the predicted number of additions to the SANs (dashed red line). In our dataset of 315,796 websites we change only the certificate for the website visited. From the SANs that changed, Figure 6.8 shows a median shift from 2 SAN DNS names to 3. The shift increases at the 75th percentile from 3 to 7. The distribution above the 94th percentile has a long tail, as shown in the inset.

[2]https://letsencrypt.org/stats/

What is the scale of required changes?

The distribution of entries in the SAN is independent of the scale of change to each individual certificate, which we plot in Figure 6.9 ranked by the size of the SAN (in red). For each certificate the corresponding size of the SAN change is plotted in green, and the ranked order of predicted size after the changes in blue. In our dataset, 195693 (62.41%) of the 315,796 website certificates require no modifications, as represented at logscale in the tail of the figure. With no more than 10 changes, 290509 (92.66%) of websites are able to coalesce additional hostnames. We also identify long tail effects where ≈1% (3129) of the websites need more than 78 new DNS entries to their certificates. Our analysis finds that 230 websites already serve certificates with more than 250 DNS names in their certificates' SAN indicated by the red dots on the scatter plot. By updating existing certificates, we identify that this number increases by 130% to 529 websites which now serve certificates with more than 250 DNS SAN names with the largest certificate in our analysis now containing 1951 DNS SAN names indicated by the blue dots in Figure 6.9.

A detailed ranking of the top-10 measured and predicted SAN sizes is presented in Table 6.7a. For example, the top-ranked measured and predicted SAN size of 2 is unchanged. Interestingly, there is only one change in the SAN sizes that rank in the top-10: From the measured ranking, size 10 SAN entries drop out are replaced by size 7 — otherwise the set of sizes in the top-9 is unchanged. Beyond view from Table 6.7a, we predict that 259315 (81.94%) websites would have up to 11 DNS SAN names in their modified certificates, which indicates minimal changes for the majority of the websites. While large certificates do reduce the number of certificates needing validation, they have additional implications on performance. In our dataset there are also 11,131 websites with no SAN names in the certificate. Among those, only two certificates need changes to include coalescable hostnames. The remaining 11,129 (99.98%) websites serve their subresources, or have no coalescable hostnames, or both. They account for ~3% of the total websites in the study and can be seen in Figure 6.8 at $x = 0$.

Rank	#DNS SAN Entries				Rank Change
	Measured	Count	Ideal	Count (Pct. Change)	
1	2	143037	2	104604 (-26.86%)	=
2	3	73124	3	46772 (-36.03%)	=
3	1	30278	4	23701 (-21.72%)	↑3
4	0	11131	5	20130 (+80.84%)	↑5
5	8	8343	8	12408 (+48.72%)	=
6	4	7223	1	11456 (+50.60%)	↓3
7	9	6380	6	11297 (+77.07%)	↑1
8	6	4141	0	11129 (+168.75%)	↓4
9	5	3149	9	9833 (+212.25%)	↓2
10	10	2573	7	9295 (+261.25%)	↑2

(a) Distribution of the total SAN names in certificates after addition of DNS SAN names to certificates.

Provider	#Sites	Hostname	Count	%
Cloudflare (AS 13335)	78155 (24.74%)	cdnjs.cloudflare.com	12675	16.21
		*.cdnjs.cloudflare.com	12675	16.21
		sni.cloudflaressl.com	9839	12.58
		ajax.cloudflare.com	8816	11.28
		cdn.jsdelivr.net	6793	8.69
Amazon-02 (AS 16509)	24494 (7.75%)	cloudfront.net	4907	20.03
		*.cloudfront.net	4907	20.03
		*.hotjar.com	3620	14.77
		hotjar.com	3613	14.75
		*.s3.amazonaws.com	2944	12.01
Google (AS 15169)	16078 (5.09%)	google-analytics.com	13776	85.68
		*.google-analytics.com	13776	85.68
		urchin.com	13776	85.68
		www.googletagmanager.com	13300	82.72
		fps.goog	13300	82.72

(b) Top 5 frequently requested hostnames to include in certificates issued by Top 3 hosting and certificate providers

Table 6.7: Changes required in served certificates and breakdown of most impactful hostname additions to certificates by provider in the short-term.

Most Effective Changes

Our analysis identifies popular cloud providers and CDN services that can provision certificates for their users. These certificates can be modified to include other subresources offered by the same CDN that serves the website. We identify individual providers and, alongside, the most frequently used hostnames for subresources in our dataset. For example, Table 6.7b shows the top three providers (Cloudflare, Amazon, and Google) in our dataset collectively responsible for serving 37.5% of the websites. Each entry also lists the five most used domains for those websites' subresources available at the same provider.

Among this set 78,155 websites in our snapshot are served by Cloudflare, comprising 24.74% of the websites measured, followed by Amazon Web Services (7.75%) and Google (5.09%). The most used domain for subresources is cdnjs.cloudflare.com, which is requested by 16.21% of websites served by Cloudflare. These webpages currently incur network and endhost resources for DNS and TLS that could otherwise be eliminated by coalescing if the hostnames were included in all

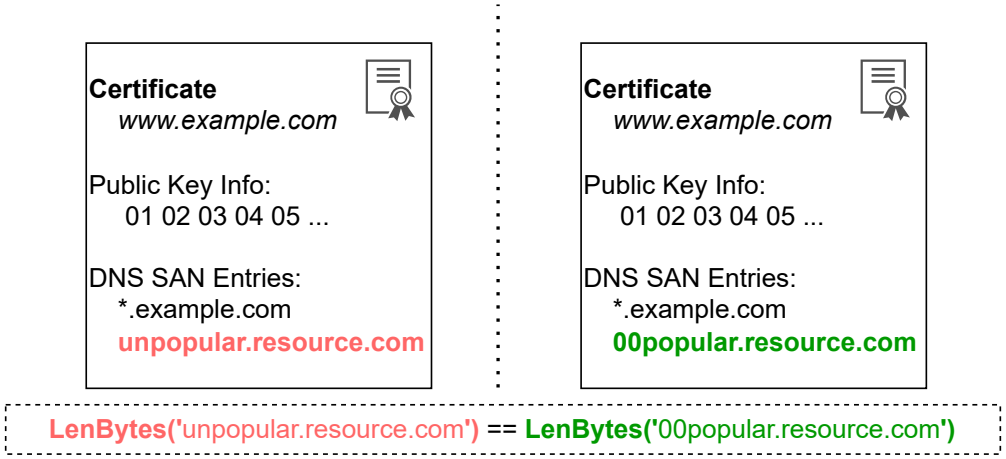


Figure 6.10: Experiment Setup - Certificate Issuance

Cloudflare issued certificates. Similarly, cloudfront.net has multiple sub resources that are used by 20.03% of Amazon users and the inclusion of the hostname in the DNS SAN of the certificates would improve clients' ability to coalesce connections. The inclusion of google-analytics hostnames in certificates from Google resources would result in improved connections management and reuse of connection from 85.68% of webpages served by Google.

Observations summarized by Table 6.7b suggest that large gains await with least-effort by adding a set of most-used subresource hostnames to existing certificates, and enabling ORIGIN Frame. It behooves large providers, in particular, to consider adding their most-used hostnames to certificates by default, and implementing support for ORIGIN Frame. On the client-side, since no certificate changes are needed, only ORIGIN Frame support is needed. In the absence of support, clients must fail-open according to the specification by ignoring the ORIGIN Frame [50, 330].

6.2.4 Validation with Production Traffic

To validate the correctness of the modeling, we partner with Cloudflare, a large CDN network serving a significant number of websites, and deploy two experiments months apart.

Sample Group and Certificates' Setup

Our deployments coalesce a third party (i.e. subresources) domain, hosted by the same CDN, that is used by ~50% of the top 1M websites [328], and receives over 5 Billion daily requests, worldwide [58]. Due to the importance of this third party domain, its request pipeline is provisioned with high-level traffic-engineering and service-level agreements (SLA). Notably, this means that changes to either the IP address or the request pipeline has implications for the CDN's wider service, and necessarily shapes experimental designs. For example, candidate domains were restricted to lower-tier service-level agreements (SLAs) to mitigate risk.

Our initial sample set was the 5000 domains with the most requests to the third party, as indicated by the Referer field in the CDN's third party logs. To respect privacy, the Referer field was truncated at the domain, and omitted subpages in the URL. From this set we found and removed 22% of websites that could not trigger the required actions from the active measurements because only their subpages request the third party resources we attempt to observe the connection coalescing to. Each domain was randomly assigned into an experimental or control group.

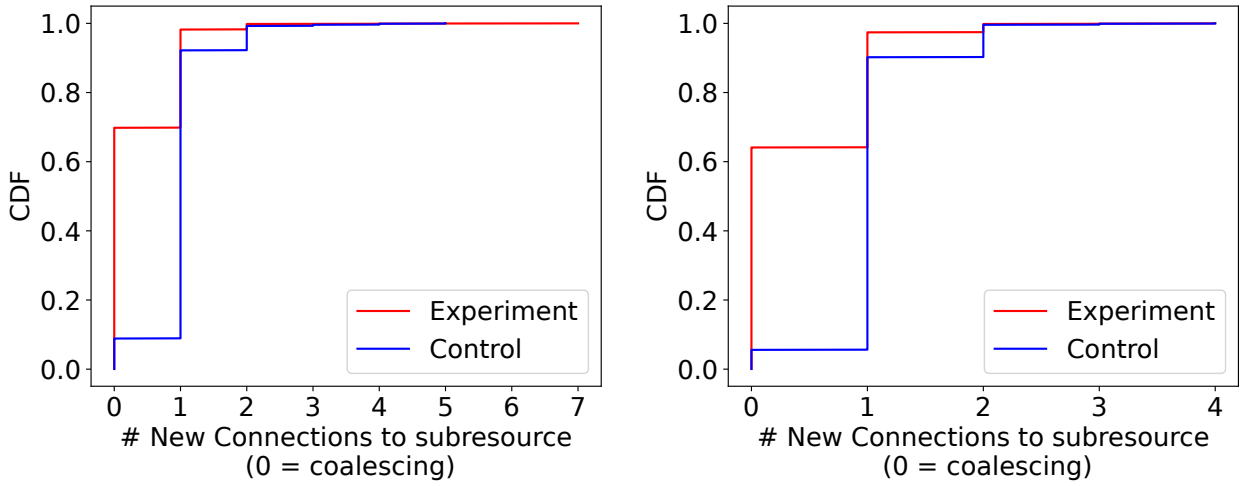
Recall that both IP and ORIGIN based coalescing require all coalescable domains to appear in the certificate SAN. We modified all existing certificates for the 5000 domains served by the CDN, as depicted in Figure 6.10. For the experimental group, the corresponding certificates were renewed with the third party domain (20 bytes) added to the SAN. To ensure integrity between control and experimental sets, we selected a valid and identical size third party domain used by *none* of the control domains. The certificates were also renewed for these domains to include the unused domain. As a result all certificate modifications in both treatment groups consisted of the same number of byte additions.

Evaluating IP Coalescing

In addition to certificate changes, IP coalescing requires that (i) DNS returns the same IP address for domains to which connections could be coalesced, and that (ii) the webserver can respond to and satisfy content requests for all those domains, on that IP address. In our experiment we elected to use one single address for all 5000 domains in the experiment and for the two third party domains. The necessary changes were facilitated by mechanisms supporting addressing agility [126]. One operational challenge in this deployment was that the SLA associated with the sample group was very different from the SLA for the two third party domains. We limited exposure by (i) deploying to two medium-sized datacenters, and (ii) using a new and unallocated IP address. Corresponding changes were made to DNS to respond with that address, and to web servers to accept connections, for the control group and third party domain. Web servers were additionally configured to service requests where the HTTP Host for the third party domain was different from the SNI in the TLS connection (to pass domain-fronting checks). The deployment duration was two weeks in August 2021.

Server-side Passive Measurement

A randomly sampled 1% of HTTP requests were logged. Requests to the third party domain from the experiment and control groups were tracked by the Referer field. In the default logging pipeline all requests in a TLS connection were with a unique identifier that, coupled with the referrer, was sufficient to count non-coalesced requests to the third party. However, there exists no obvious flag, field, or marker in TLS nor HTTP that would indicate a request is in a coalesced connection. Therefore, we modified the pipeline to additionally (i) set a flag bit for requests when the HTTP Host differed from the TLS SNI, (ii) treatment label (experiment/control), and (iii) label each request with its arrival order in the connection. We take the flag bit set as a reasonable signal of connection coalescing. From these requests, we look for arrivals ≥ 2, making sure to count the



(a) IP-Based Coalescing

(b) ORIGIN Frame Coalescing

Figure 6.11: Connection reuse seen through the reduction in the number of TLS connections made during page load events as a part of active measurement.

corresponding unique identifier only once. Requests to third party domain from the experiment and control treatments groups were confirmed to have similar profiles (request rate, popularity). Across all browsers and client requests to the websites in the two treatment groups in the study, we observed a 56% reduction in the rate of TLS connections to the third party domain from the experimental group relative to the control group.

Client-side Active Measurement

If connections are being coalesced we would expect to see zero new TLS connections from the client to the third party. In our experiments, the number of new connections are captured by Figure 6.11a. Results shown are for Firefox (v91) for later comparison, since only Firefox has client-side support for ORIGIN Frame. In the control group, the proportion of 0 and 1 new connections is approximately 9% and 83%, respectively; and overall 90% making between 1 to 5 new connections. None of the control group initiated more than 7 new connections to the third party domain. In the

experiment group, approximately 70% of visits trigger no new connections at all. A further 28% make one new connection, and no website uses more than 4 new connections.

Coalescing support via IP addresses needs alignment in both DNS responses and content reachability. This creates a burden on service operators, who must ensure consistency between content and address systems that are otherwise unrelated. However, their separation is needed to help maintain SLAs and different request pipeline optimizations for different domains or traffic types. ORIGIN Frames diminish these burdens on the operator.

6.2.5 Evaluating ORIGIN Frame Coalescing

Deployment Setup

ORIGIN Frame removes the need to align IP addresses for different domains, and reduces operational costs and complexities on CDNs. On balance with the nominal changes to certificates predicted by our model, the maintenance burden of ORIGIN Frames is isolated from the alternative, and preferable. IP-based coalescing, by comparison, requires co-orchestration with DNS and name-to-IP mappings, which reduces an operators ability to provision their systems or meet SLAs. In addition, misconfiguration risks hostnames being unreachable. In contrast, the consequence of a misconfigured ORIGIN Frame, for example with unreachable names, is to cause clients to 'fail-open' and revert to normal behaviour without coalescing. Unfortunately, there exists no production-grade implementation of ORIGIN Frames in HTTP/2 web servers. To mitigate risk on the CDN's request pipeline, we implemented and integrated a custom connection-termination process, with ORIGIN support, into the production environment [88, 90].

Our custom process was configured to accept and service requests for the sample group. ORIGIN Frames were populated with either the third party or control domain to match the sample's certificate. Finally, the third party DNS changes were undone, which immediately restored the CDN operator to its standard traffic-engineering practices and SLAs. Unlike the regional-restrictions

needed to minimize risk with IP-based coalescing, the ORIGIN Frame implementation was deployed globally at over 275 points of presence [91]. To facilitate observability, the sample group was moved onto an isolated anycast address. Doing so meant that the CDN's network operations could, if needed, shift experimental traffic away from a datacenter by withdrawing the corresponding IP prefix from BGP announcements.

Finally, the decisions to implement a custom process and deploy globally were taken only after we could test and confirm that ORIGIN is either ignored or handled correctly by in-support browser versions. The deployment duration was two weeks in January 2022.

Server-side Passive Measurement

The same logging pipe-line, 1% random sampling, and signals described in during the IP based coalescing measurement were hardened for global deployment, and used to count coalescing with ORIGIN Frames. Global data was further filtered by user-agents corresponding to Firefox, the only browser at the time of deployment with support for ORIGIN. The daily number of new TLS connections for the sample group are shown longitudinally in Figure 6.12. The daily new TLS connections from the experimental group to the third party initiate approximately half as many new TLS connections compared to the control, and relative to daily counts before and after testing. Only in our active measurements did we discover that further coalescing was obstructed by use of the HTML `crossorigin` attribute set to anonymous in subrequests providing support for CORS [341].

Client-side Active Measurement

The number of new connections under ORIGIN using Firefox (v96) are captured by Figure 6.11b. In the control group, the proportion of 0 and 1 new connections is approximately 6% and 84%, respectively; and overall 94% making 1 to 4 new connections. In the experiment group, approxi-

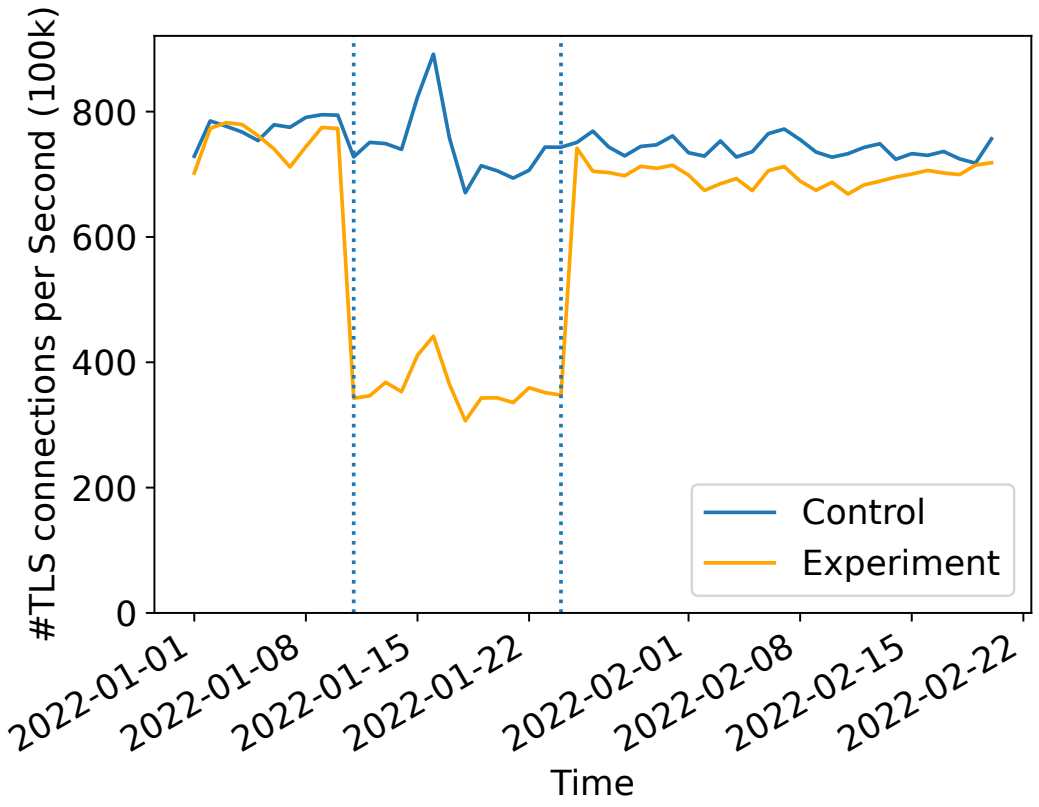


Figure 6.12: Reduction in Number of TLS connections made to coalesced sub resource observed during the duration of the experimental treatment

mately 64% of visits trigger no new connections at all. A further 33% make one new connection. As a general improvement over natural IP coalescing, none of the sample set visits make more than 4 new connections to the third party domain with ORIGIN enabled, which is less than both groups made with IP coalescing, alone. While we expected higher percentages of connections to be coalesced with the usage of ORIGIN Frames (Figure 6.11b) compared to IP based coaslecing (Figure 6.11a), the lower percentage observed in our active measurements could be explained by the churn in resources required by the websites in the study due to modifications to the websites in our study by their developers given the time elapsed between both the experimental setups (August 2021 and January 2022), possible sampling and artifact biases.

In this deployment we also perform a longitudinal view of new TLS connections to the third party from websites during the deployment of ORIGIN Frames, shown in Figure 6.12. The difference in new connections initiated by control and experimental groups during the deployment is in stark contrast to new connections initiated before and after. The average reduction during the experiment is clear at ≈50%, but also less than expected. Upon later inspection of websites in the experimental group, we observed the usage of XMLHttpRequest and fetch to make requests to the subresource which do not coalesce connections, similar to subresources obtained with a value *anonymous* for the HTML attribute crossorigin.

6.2.6 Performance, Correctness, Trust, and Privacy Implications

The results indicate that connection coalescing through ORIGIN frames can significantly improve the rates of connection re-use. This has implications for both infrastructure operators and the web ecosystems.

Connection coalescing opens resource scheduling opportunities. Re-ordering and restructuring of webpages has been shown to greatly affect page loads [190, 345]. However, the sequence of resources transmitted over multiple connections may be altered by network effects, and received by the client with different ordering and timings. An unintended ordering may then delay receipt of higher priority objects on a parallel connection, or contribute to head-of-line blocking in the application. In contrast, coalesced resources are always received in the ordering intended to optimize the critical path [343].

On performance in general, we *emphatically refrain from 'faster' as an assumed outcome or primary motivation.* Our preliminary evidence suggests 'no worse' is appropriate. The subtle interplay between resource object sizes, competing connections, and congestion control [12] is subject to network conditions. Bottleneck-share capacity, for example, diminishes as fewer connections compete for bottleneck resources. Yet many pipelined small objects may favour bytes

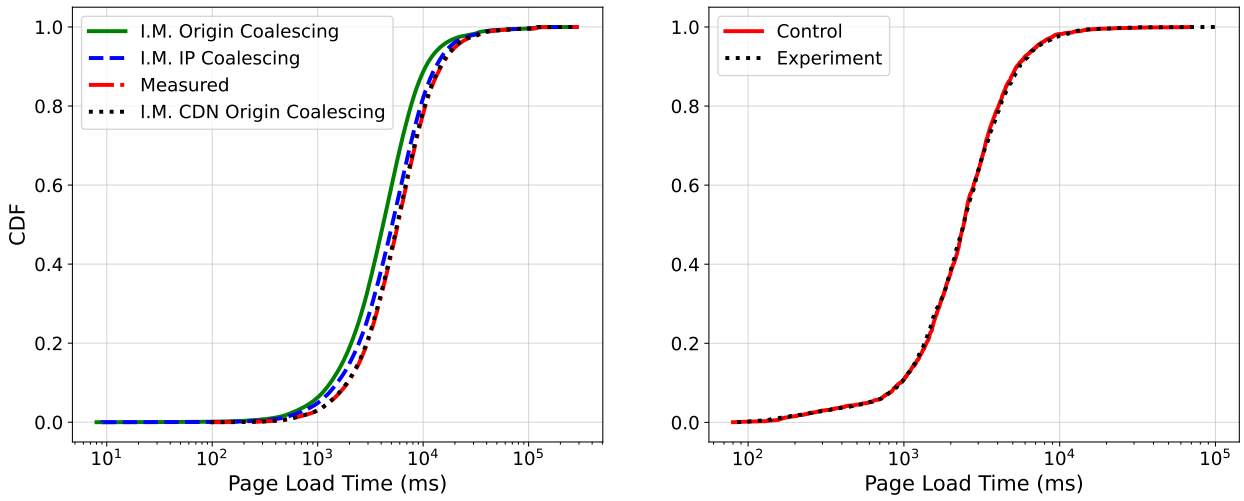


Figure 6.13: (*Left*) Model predictions of connection coalescing impact on Page Load Times (PLT); dotted-black line predicts only the deployment CDN. (*Right*) Measured PLTs at the deployment CDN with ORIGIN support indicate potentially minor improvement, but 'no worse' performance. (I.M = Ideal Modelled)

transmitted in steady state on one connection, over bytes delayed by slow-start and discovery over many connections. Additionally, outside of bottlenecks, web interactions are dominated by latency over goodput [30, 131, 299]. This position is reinforced by our own attempts to demonstrate improvements in page load times (PLT), as captured in Figure 6.13. Modelling on our dataset as shown in (top) Figure 6.13, suggests that IP based (dashed blue line) and ORIGIN (solid green line) coalescing would improve median PLT by ~10% and ~27%, respectively. These predictions establish an upper bound for ideal conditions in which every server or service provider fully enables IP or ORIGIN Frame support for connection coalescing as indicated by our model. However, the dotted black line indicates the performance we measured with only one CDN implementing ORIGIN frames and coalescing *one* popular host. The improvement of 1% indicates that the model is fairly accurate but is too small in the limited scope of the measurement performed. We strongly believe that with engineering changes in both browsers and CDN networks coalescing multiple hostnames, we will be able to see better performance.

We emphasize, however, that while coalescing may be equally achieved via IP or ORIGIN, operator complexities make the former difficult to support.

Browser Trust Models: Browsers can trust existing connections for hostnames in the ORIGIN set with valid entries in the certificate's SANs due to the existing trust model where only the certificate served on a connection is used to validate the correctness of the hostname being communicated with. As a result, any changes to ORIGIN frames require certificate changes which could otherwise have been implemented through upcoming proposals such as secondary certificates [57]. The proposed changes to certificates introduce an ORIGIN Frame equivalent called *certificate frames*. These are blocking stream operations sent in stream 0 and provide alternative ways to obtain server certificates other than through the initial TLS handshake. If needed, large certificates can be broken into smaller certificates to match the TLS record sizes and sent using the proposed certificate frames. However, the usage of certificate frames incurs the transmission of complete X.509 certificates by the servers each with embedded public key, and signature which are the largest segments of the certificate increasing their size compared to the required certificate modification. However, the certificate frames could provide some flexibility to service providers to send certificates on demand without modifying the main certificate of the webpage or including the popular DNS name in multiple certificates, and could benefit from additional investigation.

Today, browser client support for ORIGIN frames exists only through Firefox. Other clients may be discouraged by the lack of clarity in the CORS standard [341]. Wider adoption may hinge on refinement or instruction in the Fetch standard and Same-Origin Policy [340]. The Online Certificate Status Protocol (OCSP) is independent, but notification of certificate revocation may give further confidence in the certificate's validity without DNS [292].

Privacy Improvements: While the question of whether ORIGIN frames facilitate or challenge fingerprinting techniques is an open question, it is important to observe that each coalesced connection reduces the need for making additional DNS queries (which are blocking in nature),

and hides an otherwise exposed plaintext SNI from the TLS handshake. Majority of users today continue to use plaintext DNS queries reduction of which reduces the ability of fingerprinting services on the network from understanding user's traffic pattern. The on-going colocation of services and serving millions of hostnames from a single IP address, adds to the challenges for fingerprinting services, providing users more privacy – an advantage provided by the co-location of services among a few select operators, and the reduction of otherwise plaintext queries.

Web Packaging and Proxy Based Accelerators: An alternative approach which is actively being considered by organizations and standards bodies is the usage of web packaging signed HTTP exchanges [355]. These packaging bundles allow content to be decoupled from origin servers and served by intermediate nodes without losing the cryptographic guarantees of connecting to the origin and behaves equivalent to scenarios with complete connection coalescing. Efforts are ongoing to improve privacy with the increased usage of web package signed HTTP bundles using private prefetch proxies [65]. These approaches while potentially improving performance beyond ORIGIN Frames pose privacy, and fairness challenges by preventing traffic from actually reaching the origin servers. This has been actively adopted by news websites using Accelerated Mobile Pages (AMP) but has also been heavily criticized [21].

Low Impact on Certificate Transparency: Certificate authorities (CAs) issuing certificates to a website for a hostname also write the certificate to multiple CT logs, operated by different organizations and monitored publicly [203, 210]. Modifying 37.59% (120103) of the websites contributes to a 5-10% increase in the number of certificates issued by CAs daily [7, 89]. The unbalanced publishing of certificate records between CAs and CT log operators results in increased stress among a few large CT log operators (eg. Cloudflare, Google) but can be mitigated by improving load distribution among log operators [296]. The global certificate issuance rate is 257034 certificates per hour indicating that a bursty one time operation of certificate modifications would not adversely affect CT log infrastructure performance [89].

Non-compliant HTTP/2 Software Stacks: An unanticipated outcome of the deployment of ORIGIN frames is that it exposed non-compliant HTTP/2 implementations. The HTTP/2 specification mandates that clients ignore and discard unknown frames [50], a mandate respected by all operating systems and supported browsers. However client communications may be subject to separate network stacks running, for example, antivirus software with Internet security features, or corporate firewalls, and networks that install custom managed self-signed certificates. During our experiments with ORIGIN Frame, a developer of anti-virus and Internet security software products contacted the CDN to enquire about an increased number of failed connections to the websites in our study. Following the rules of disclosure, a collaborative diagnosis pinpointed the issue to an unknown HTTP/2 frames handling bug in the developer's network agent. Rather than ignore the unrecognized frame as is required by the specification, the network agent instead would tear down the TLS connection and prevent clients from accessing the websites in our experiments. Upon careful consideration by the CDN, and given the sensitive nature of the antivirus software developers' operations, the CDN agreed to limit disclosure and pause ORIGIN testing for a limited duration. In September 2022, the antivirus provider confirmed to the CDN that the issue in the product had been fixed.

This indicates one of possibly other unintended consequences due to the deployment of privacy and performance enhancing mechanisms on today's Internet. In this work, we identify opportunities for operators that host third party resources. With ORIGIN Frame, our models suggest that adding the most popular of those domains to the appropriate certificates (see Table 6.7b) would reduce the number of TLS connections made to the resources, thus improving server compute overheads and benefiting clients that can and choose to coalesce their connections. Through our evaluations, we confirmed that the Firefox browser conservatively continues to make new and subrequest *blocking* DNS requests to hostnames in the ORIGIN Frame, despite their inclusion in the modified TLS certificate. These additional queries could be avoided, conferring privacy and

other benefits to users. Doing so does not alter isolation techniques, for example, as exists in the render process to prevent security attacks [279]. For improved adoption, we also recommend web server software to integrate support for configuring ORIGIN Frames. As a part of this work, we open source our ORIGIN-supporting changes to the Go lang $\mathrm{net/http}$ implementation [88, 90].

6.2.7 Conclusion

ORIGIN Frames are extremely useful hints when respected by browsers and need server modifications. Our experiments reveal an important role that ORIGIN Frames could play in today's Internet. Our analysis on a large-scale dataset finds that adding no more than 10 DNS names to 37.59% of the certificates will reduce certificate validations (i.e. new TLS handshakes) by 68.75%, while reducing the number of render blocking DNS queries by 64.28%. Clients additionally reap these benefits in privacy by reducing cleartext DNS exposure to network on-lookers. Our measurements, and deployments at scale indicate that these proposed changes are feasible and offer a glimpse into the potential of ORIGIN frames towards improving privacy, experienced performance by the users and optimizes resource utilization at the network operators by reducing potential overprovisioning requirements.

Chapter 7

Building Trustworthy and Secure Applications

7.1 Reducing Trust Placed In DNS Recursive Resolvers

With mechanisms to improve privacy, and reduce trust in hidden DNS infrastructure on the Internet, a key problem which remains is the trust placed in the *correctness* and validity of the responses obtained by users from their resolvers. With the usage of encrypted protocols, large network providers with extensive IP address allocation space, can respond with a unique IP address to each query and trace the follow up TCP and TLS connections from the user to the returned IP address. While ODoH guarantees privacy within the scope of DNS queries, these guarantees do not apply for network communications outside of the DNS ecosystem. The deployment of ORIGIN frames while addresses some of these concerns, still poses the risks for unique user identification resulting in loss of privacy. Similarly, a typical Internet user does not manually configure secure DNS protocols, and relies on the configuration and infrastructure provided by their Internet service provider or carrier. Despite securing the communication links, there exists a risk that the resolver itself could tamper and change the actual response for the intended query.

A malicious resolver returning the responses for a client query for a website might return IP addresses which do not belong to the website but rather an intercepting middlebox attempting to impersonate the resource the client is intending to navigate to. This technique is used by many

ISPs and carrier networks deploying firewalls to enforce content blocking due to court mandated orders, or due to government mandated censorship. While the legitimacy of DNS *censorship* ([172, 265]) of this nature is questionable [122, 334], it opens up questions about the correctness and honesty of the recursive resolvers. Despite the security provided by the transport layer protocol, clients continue to be vulnerable to resolver lies since having a safe channel with a lying resolver does not prevent the client being returned invalid data.

As communications between the various entities of the DNS ecosystem are being secured, other attacks targeting the integrity and availability of the responses continued [27, 153, 179, 188, 236, 303, 315]. While mechanisms addressing these attacks such as DNSCurve, and key pinning were proposed [19], DNS Security Extensions (DNSSEC) providing integrity and authenticity guarantees for the responses from DNS servers was eventually standardized [286–288]. Despite standardization, the adoption for DNSSEC remains low with arguments made against it due to deployment complexity, increased message sizes, and compute burdens resulting in denial of service attacks [96, 152, 167].

Today, though clients can perform DNS queries requesting DNSSEC related information from the resolver and validate them, most default client devices do not request this information by default and instead it must be explicitly enabled by the user [93, 198, 206, 212]. Over 95% of DNS requests received at a large public DNS resolver operated by Cloudflare do not request DNSSEC enabled responses. By design, to perform DNSSEC validation, clients must make all the required queries for the relevant keys and validate the cryptographic bindings to the parent zone through additional queries. We described this approach previously in §2.2 and present the traversal of the hierarchy in Figure 2.3.

Every DNSSEC validating client making queries to the nameservers directly has the adverse effect of increasing the client's networking overheads, and in-turn the request load on the nameservers, thus requiring significant human administration for scaling and reliability. Validating

recursive resolvers purport to ease this burden by performing the DNS resolutions themselves and validating necessary DNSSEC records. However, this leaves a few things wanting: 1. it may not be sufficient for the client to trust that the resolver validated the DNSSEC records correctly, 2. resolver's DNS cache causes a tradeoff between resolution speed and caching inconsistency due to upstream changes, and 3. clients may not have the bandwidth or infrastructure for increased queries. Well known public resolvers such as Google, Cloudflare, NextDNS etc., operate DNS services and validate DNSSEC responses before issuing a response to the client query. However, DNS resolvers are typically intended to provide their services to clients within a private network (typically ISPs or organizations) allowing the clients in the network to benefit from the advantages of shared caches, reducing the network load on authoritative nameservers. Prior research has identified that these ostensibly internal resolvers are sometimes publicly accessible and provide DNS resolution capabilities to unprivileged users on the Internet [196, 197]. These open resolvers enable adversaries to perform distributed denial of service (DDoS) through *reflection* or *amplification* attacks [15, 20, 178].

Despite the prevalence of open resolvers on the Internet and the importance of DNSSEC, the correctness of DNSSEC aware resolvers remains unmeasured. To motivate the need for clients to validate the obtained DNS responses, we perform Internet measurements to identify DNS resolvers and their ability to validate DNSSEC queries. We observe that majority of the resolvers on the Internet today (75%) respond to DNSSEC queries in ways violating Internet standards [287] – indicating the need for clients to perform their own validation without fully relying on their network configured DNS resolver. Clients are automatically configured local DNS resolvers when connecting to public WiFi hotspots at airports, coffee shops, etc.., where manual trusted public DNS resolver configurations may fail, leaving the client disconnected from the Internet.

To address some of these challenges around the correctness of responses and in an effort to reduce the implicit trust in the DNS resolver, we take the position that the DNSSEC ecosystem

would benefit from a simple change to the recursive resolver — serializing the individual DNSSEC records and including them in client responses. This type of scheme has both performance benefits for security-conscious clients, is highly practical to deploy requiring changes only to recursive resolvers in the overall DNS ecosystem, and could unlock a host of new applications leveraging the DNS ecosystem. It efficiently brings DNSSEC to clients by proposing that recursive resolvers provide *all* required DNSSEC records if requested. This would be signaled by reusing a DNSSEC single bit flag in client DNS requests. The response would include the regular answer in the answer field, as expected, but would also include an entire compressed DNSSEC proof chain in the Additional section of the resolver DNS response allowing clients to validate the attached proof chains with necessary client DNS stub software updates [232]. The inclusion of the serialized DNS responses in the additional section of the DNS response does not conflict with existing DNS client behavior which can discard the information – a design advantage allowing resolvers to immediately deploy the scheme without adverse impacts on clients.

7.1.1 Understanding DNS Response Headers and Message Structures

A DNS message is a packet structure for holding *resource records* (RRs), which are mappings between hostnames and different types of resources. The DNS message itself contains four sections of RRs and is serialized and sent over the wire between the clients and servers. (1) The *Question* section includes the query and the associated query parameters, (2) the *Answer* section contains the response to the query and is set by the responding resolver or name servers, (3) the *Authority* section describes additional information about authoritative name servers and can include optional information related to the answer, and (4) the *Additional* section which can contain optional but helpful RRs such as glue records, or answers for any predicted future queries allowing cache pre-population.

The DNS message also includes metadata in the header such as a 16-bit identifier, a 4-byte

integer response code (RCODE), and an OpCode in addition to a series of bit flags [64]. A single bit flag is used to distinguish the DNS message between a *query*, and a *response*. We detail the correctness conditions for other supported bit flags below.

1. **Authoritative Answer (AA)**: The AA flag indicates that an Authoritative Answer was set by the authoritative name server of the query and *must not* be set by the resolvers or the clients making the query [232].

2. **Recursion Available (RA)**: The RA flag indicates the ability of the resolver or the name server to recursively perform a resolution of a given query for the client. DNS resolvers accessible over the Internet to other clients set this bit in their response indicating their capability [232].

3. **Recursion Desired (RD)**: The RD flag is set by the client making a query to a recursive resolver [232]. The RD bit being set indicates the client's request to the server to recursively resolve the client query, while the flag not being set indicates the client's intentions to perform the recursive resolution on its own. A correctly configured resolver copies the value set by clients into the response.

4. **DNSSEC OK (DO)**: A client query with the DO bit set indicates the client is requesting the associated DNSSEC signatures in addition to the answer for a given query [25]. The bit is also set by a validating resolver for all the responses sent to the client that include the requested signatures. Validating resolvers strip out the signature records when the flag is not set by the client preventing unnecessary data being sent over the network and reducing the message size in transit.

5. **Authenticated Data (AD)**: The AD flag is set by a validating and DNSSEC-aware resolver after successfully performing the required validation procedure (as described in RFC

4035 [287]) on the RRs acquired from the recursed name servers. The AD bit is not set when directly obtaining an answer from an authoritative nameserver.

6. **Checking Disabled (CD)**: The CD flag is used for DNSSEC-based responses to instruct the resolver to return any responses leaving the onus for validation on the requesting client. A validating resolver continues to validate the response and can copy the CD bit in addition to appropriately setting the correct AD bit.

Critically, when a client queries for a hostname with a broken or unavailable DNSSEC enrollment with the DO bit set and the CD bit unset, a correctly operating and validating recursive resolver should never set the AA, AD, or CD flags in the response, and set SERVFAIL response code (RCODE=2).

7.1.2 DNSSEC Is Undervalued

DNSSEC has seen slow adoption since its proposal in 1997 [5]. Measurements done by Wander in 2017 indicated less than 1% DNSSEC adoption among popular TLDs [342]. Chung *et al.* showed in the same year that as few as 0.6% of com. domains had DNSKEYs published—necessary though not sufficient for successful DNSSEC deployment. One reason for slow DNSSEC adoption is the perception that it offers only marginal benefits in the presence of TLS certificates since resolution integrity becomes obvious when presented with a server certificate. A valid certificate implies that the underlying resolution must have been correct. However, there are still a number of important use cases that TLS certificates do not address. Foremost among these concerns is the issuance process of TLS certificates themselves. Certificate Authorities (CAs) only issue certificates after confirming the domain ownership through a *domain validation* (DV) process. The success of a system addressing the challenges of trust and misbehavior depends on the deployment of DNSSEC in the DNS ecosystem.

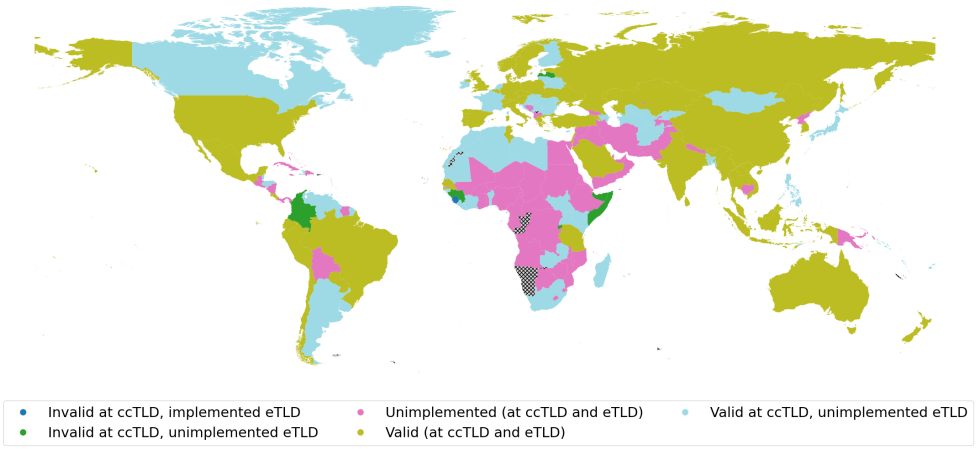


Figure 7.1: The categorization of DNSSEC implementation at ccTLD (e.g. au. and .br) and eTLD (e.g. gov.au. and gov.br) levels per country. For countries with multiple government eTLDs *(eg. .gov.us and .gov)* most popular is chosen.

We compare the adoption of DNSSEC at the ccTLD level (eg. *.uk*), with that of the respective country government eg. *.gov.uk* due to their critical nature for information dissemination on the Internet – which we term as the *eTLD* for this section and present a map as shown in Figure 7.1 to understand the adoption of DNSSEC at each country and their respective governments' host-names [199]. We observe 57 countries (light green) have valid DNSSEC enrollments, 68 (light blue) have DNSSEC enabled at the TLD but not at the government eTLD, 68 others (magenta) with the majority in Africa and the Arabian region have not enabled DNSSEC at either level, and 5 countries (dark green) have invalid DNSSEC entries at the ccTLD. Sierra Leone is the only country which has invalid DNSSEC ccTLD entries but attempts to use DNSSEC for their eTLDs.

Let's Encrypt famously automated certificate issuance through DV in addition to providing several alternate validation methods [8, 47]. In their *DNS challenge* method, the ownership of the domain is established by adding a random value associated with the challenge inside a DNS TXT record. If the CA is able to retrieve the value via DNS, they have a strong assurance that the domain is owned by the claimant.

For domain owners who wish to be provisioned a wildcard certificate, or with multiple SAN entries in the certificate as required today and support ORIGIN frames, DNS challenge is the *only* accepted method. In the 90 days preceding September 22, 2019, Let's Encrypt validated 266M domains, 13% of which used the DNS challenge method; and continues to issue millions of certificates everyday [8, 89]. In 2018, Brandt *et al.* were able to acquire a certificate for a domain they did not own by exploiting weaknesses in unprotected DNS. Since then, some CAs like Let's Encrypt enforce the verification of the DNS challenge through multi-vantage point checks which make it hard but not impossible for attacks (such as BGP hijacking) to attack DNS validation schemes [55]. Additionally, CAs increasingly rely on validating DNS recursive resolvers as a part of the challenge validation process sometimes even relying on large public open recursive resolvers as the upstream resolvers to prevent managing their own DNS resolver infrastructure [55]. While these make such attacks significantly harder, they do not entirely preclude them as DNSSEC would.

In a similar vein, the use of HTTPS records is becoming more common for specifying various aspects of HTTPS services such as TLS EncryptedClientHello [283]. Relying on the resolvers and neglecting the integrity of these records leaves clients vulnerable to surveillance through MITM or downgrade attacks, especially as new regulations mandate the use of country-specific DoH resolvers – intended to enforce content policy [29, 268]. Decentralized Web3 ecosystems such as the InterPlanetary File System (IPFS) rely on signed DNS TXT records [109], while the Ethereum Name Service (ENS) also uses similar records to prove domain ownership and mandate DNSSEC usage [124]. The security of communications and policy enforcement in the email ecosystem through protocols such as DomainKeys Identified Mail (DKIM) [194], Domain-based Message Authentication Reporting and Conformance (DMARC) [195], and Sender Policy Framework (SPF) [186] rely on the correctness and integrity of DNS resolvers to retrieve stored policy and key materials for preventing spam & abuse.

While the TLS-based CA ecosystem has been widely accepted, there exist alternate proposals such as DNS-based Authentication of Named Entities (DANE) leveraging the DNS ecosystem as an alternate PKI by relying on DNSSEC [44, 114]. Though DANE has not seen widespread adoption despite being resistant to attacks such as the *compelled certificate creation attack* [316] at least partly due to DNSSEC's poor adoption, and a presumed (albeit debunked) risk of denial of service due to increased message sizes, it is used by Internet projects with strong privacy guarantees such as Tor [285].

7.1.3 Understanding Implicit Trust in DNS Infrastructure

We perform our active scan from an Amazon Web Services hosted VM c5.4xlarge VM instance with 16 vCPUs, 32 GB of memory, and up to 10 Gbps bandwidth from the US-West-2 datacenter in Oregon, United States. We configured a web server on the scanner with a static webpage informing readers of the benign nature of the scanner and included email information of the researchers to contact to opt-out of future scans and received no requests – likely because of the scan's minimal traffic and one-time nature. With the support of our collaborating CDN partner – Cloudflare, we use an authoritatively managed and well known zone brokendnssec.net, actively used for testing various DNSSEC configurations. A validating resolver attempting to resolve queries associated with this zone must result in a failure as the zone traversal encounters a cryptographically broken DNSSEC chain due to missing delegations.

Discovering DNSSEC Aware Open DNS Resolvers on the Internet

We perform an active IPv4 scan from our scanning testbed using ZMap [120] – an open source tool used for scanning large IP spaces. We capture a snapshot through our scan on December 30th, 2022. We used ZMap's UDP probe functionality to query the default UDP DNS port (53) for every IP address in the IPv4 space with the exception of private use, loopback, and reserved IP ranges as

indicated in various RFCs [61, 78, 234, 280, 349]. The DNS query was a request for the IPv4 address of google.com, a popular query, and additionally included in the list of UDP probes supported by ZMap. ZMap interprets a response from the IP addresses as DNS responses indicating the presence of a DNS resolver at the IP address and returns the list of such addresses after the scan. From the scan, we obtained 9.97 million IP addresses (9,975,490) which host DNS resolvers and are accessible over the probed port 53.

For each IP in the 9.97 million IP addresses associated with the resolvers identified in the discovery phase, we run additional probes by carefully crafting a DNS query for google.com, but with the DNSSEC OK (DO) bit set – indicating the need for DNSSEC signatures to be included in the responses and validation by the resolver. We filter out IPs that do not respond to the follow up DNSSEC queries and identify DNS resolvers accessible at 7.9 million (7,930,503) IP addresses indicating that over 2.04 million resolvers that were initially identified do not support DNSSEC queries from their clients.

To evaluate DNSSEC compliance, we query each identified DNSSEC aware resolver for the IP address associated with the brokendnssec.net zone respectively. We choose the google.com and brokendnssec.net zones because of their visibility and well-known relationship to DNSSEC. Google has not enabled DNSSEC at all for the google.com zone, so we expect to see no DNSSEC-aware bits (CD, AD) set in the responses or any associated signatures. Additionally, unless the resolver being queried is in the set of name servers operated by Google (i.e., is authoritative for answering the query), we do not expect to see the AA bit set in the responses. The *brokendnssec.net* zone managed as an experimental zone by our CDN partner, contains a multitude of DNSSEC configuration issues such as bogus DNSSEC delegation, non-existent DNSSEC public keys, and name servers which do not respond with the signatures, and thereby fails the checks done by a validating resolver [110]. As such, we expect our probe requests sent to the validating resolver to return a failure response code, or at the very least not respond with IP addresses.

We store the results for the responding server IP address, the IP addresses in the answer, and the status of various flag bits in the response header as a part of the scan.

Classifying DNS Resolver (Mis)Behavior

Our scan results indicate 98.17% of the DNSSEC aware resolvers (778584) return a valid non-errored response code for the query containing the question about Google's IPv4 addresses. The remaining return a wide variety of non-zero error codes with the majority (1.66%) returning the REFUSED code indicating no capability to recursively resolve client queries. We analyze the successful responses in various ways:

Correctness of Answers: We observe that more than 98% of the responses containing an answer contain a single IP addresses as the response to the given query. 1.5% of the responses obtained contain exactly 6 IP addresses and the remaining return a variable number of IP addresses in their response – sometimes upto 30 unique IPs. To identify if the IP addresses returned in the dataset are correct, we compare the IP addresses in the responses to those which are in the authoritative IP ranges published by Google. To do this, we construct an IP prefix tree and check that the IP addresses in the responses match any of the expected IP ranges advertised by Google [317]. Any IP address that does not match the expected IP ranges is considered to be incorrect. We suspect that open resolvers returning more IP addresses do so due to the aggregation of IP addresses obtained from the multiple name servers they communicate with. 4.08% of the resolvers in the active scan return the correct IP addresses for a google.com query. A vast majority (over 95%) of the resolvers return incorrect IP addresses in the answers that do not route to Google or match advertised IP ranges by Google[137]. The IP addresses returned match various censorship fingerprints studied previously [247], or redirect to blockpages [335].

Correctness of DNS Header Flags: A correct IP in the response does not necessarily imply overall correctness of the response due to the various bits in the DNS message header. We further attempt to evaluate the extent of non-conformity by exploring the response message headers. We observe that over 99% of resolvers which respond with an answer also correctly set the RA bit indicating their recursive capability. Meanwhile, the remaining do not set the RA bit despite providing recursive resolutions for the client query – this is incorrect behavior.

Surprisingly, despite the lack of DNSSEC on Google's zone, we observe 35.86% of the resolvers incorrectly set the AD bit claiming to have validated non-existent DNSSEC responses. Similarly, 95.29% (7.4 million resolvers) which do not belong to Google's authoritative advertised prefix ranges, incorrectly return the AA bit claiming to be authoritative nameservers. Our observations indicate only 47.06% (3.6 million) resolvers returning the IP addresses associated with the Google query correctly set both the AD and CD bits to false.

Responses for Invalid DNSSEC Zones: In addition to validating the correctness of the resolvers through analysis of header bits and answers, we also verify resolver correctness by querying for an experimental zone *brokendnssec.net*, managed by our CDN partner, which is not resolvable and should instead return an error. As mentioned previously, a correctly implemented resolver should return a SERVFAIL response and never return associated IP addresses. We observe 1.28 million IP addresses which respond to our queries – a 83.79% reduction from the 7.9 million discovered DNSSEC aware resolvers. 75.27% (923470) of these resolvers incorrectly return the answer associated with the query indicating that these resolvers do not validate DNSSEC responses correctly. Conservatively, we can conclude that the lack of responses from 6.62 million resolvers, which previously responded to the query for Google, represent reasonably *correct* behavior and are validating DNSSEC resolvers which could protect their clients from navigating and connecting to insecure services.

Unlike the case for Google queries where a majority of the IP addresses returned in the

responses were incorrect, 93.34% of the resolvers (861966) responding successfully to the experimental zone with invalid DNSSEC entries – returned a valid IP address for the query for broken DNSSEC queries. This pattern raises concerns about selective response behavior for popular DNS queries such as the ones for Google – possibly due to defaults, active manipulation attempts to block resources (as shown by the Great Firewall DNS censorship fingerprints [247, 281]), or redirects to malicious resources [196, 262, 265]. Despite receiving correct IP addresses from a majority of the resolvers, the act of sending a response *at all* is incorrect – implying that over a million responding resolvers do not correctly perform DNSSEC validations. The usage of such resolvers by users as their trusted resolvers puts them at risk of various attacks asserting the need for clients to validate their interactions with DNS infrastructure. While this preliminary exploration motivates our design, it raises questions about the reasons for such misbehavior exploring which is out of scope for this effort and left to future work.

While our active measurements indicate that many DNS resolvers accessible over the Internet return incorrect and invalid responses, it is also likely that those which are network-internal and hidden from our active probes are similarly incorrect.

7.1.4 System Design and Implementation of DNSSEC *Proof Chains*

Motivated by the necessity and increasing reliance on recursive resolvers in today's network deployment architectures, as well as a need to reduce the attack surface of critical applications, we propose a system with four high-level goals: (1) Reduce client's implicit trust in recursive resolvers, (2) Enable honest resolvers to prove to their clients that the resolutions are indeed trustworthy, (3) Ensure that there are no significant negative performance implications, and (4) Make minimal changes to existing DNS deployments causing minimal (hopefully none) disruptions when deployed at scale.

To recap, DNS *zones* due to their hierarchical and decentralized nature are maintained inde-

Algorithm 4 DNSSEC Setup at Each Zone

Input: Unsigned Zone – $\mathcal{Z}$
Output: Signed Zone – $\mathcal{Z}_\sigma$, $\mathrm{DS}_\mathcal{Z}$
$\mathcal{Z}_\sigma \leftarrow \{\}$
$\mathrm{SK}_k, KSK_\mathcal{Z} \leftarrow \mathrm{KeyGen}()$; Generating KSK Key Pair
$\mathrm{SK}_z, ZSK_\mathcal{Z} \leftarrow \mathrm{KeyGen}()$; Generating ZSK Key Pair
$\mathrm{DNSKEY}_\mathcal{Z}, \sigma_\mathcal{Z}^{\mathrm{DNSKEY}} \leftarrow \mathrm{Sign}(\{KSK_\mathcal{Z}, ZSK_\mathcal{Z}\}, \mathrm{SK}_k)$
for all $\mathrm{RR} \in \mathcal{Z}$
    $\mathrm{RR}_\sigma, \sigma_{\mathrm{RR}} \leftarrow \mathrm{Sign}(\mathrm{RR}, \mathrm{SK}_z)$
    $\mathcal{Z}_\sigma.\mathrm{Append}((\mathrm{RR}_\sigma, \sigma_{\mathrm{RR}}))$
end for
$\mathcal{Z}_\sigma.\mathrm{Append}((\mathrm{DNSKEY}_\mathcal{Z}, \sigma_\mathcal{Z}^{\mathrm{DNSKEY}}))$
$\mathrm{DS}_\mathcal{Z} \leftarrow \mathbb{H}(KSK_\mathcal{Z})$
return $\mathcal{Z}_\sigma, \mathrm{DS}_\mathcal{Z}$

pendently by various entities. A zone example.com., is the child zone of the com. zone which in-turn is a child zone of the root (·) zone. Each zone administrator generates multiple asymmetric key pairs associated with the respective zone. These key pairs are referred to as the Zone Signing Key (*ZSK*) and the Key Signing Key (*KSK*). Each zone managed by its corresponding authoritative nameserver maintains a grouped set of resource records (RR).

The *ZSK*, and *KSK* associated with the zone are grouped together as a special RR which are represented by DNSKEY. These public keys are cryptographically signed by the secret key associated with the KSK represented by SK_k and are stored in the corresponding Signed Zone file $\mathcal{Z}_\sigma$. Similarly, each individual RR group is cryptographically signed by the private key associated with the ZSK and is included into the signed zone file. With the knowledge of the DNSKEY associated with the zone $\mathcal{Z}$, a client can validate each RR and the accompanying signature RR_σ. However, the KSK used to validate the DNSKEY records are trusted in this case. To avoid this, each zone uses a one way cryptographic hash $\mathbb{H}$ such as SHA256, SHA512 or SHA1 and hashes the public KSK to generate a delegation signer (DS) record which is submitted to its parent zone for inclusion. A zone example.com. submits its DS records to *com.* which signs the DS record

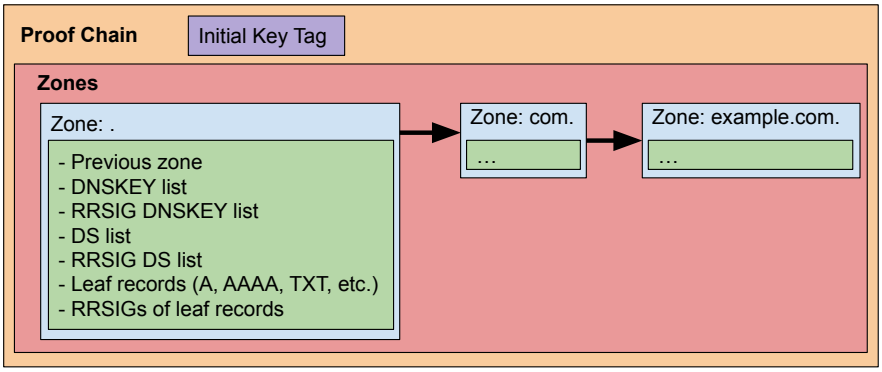


Figure 7.2: A Serialized DNSSEC Proof Chain

using its own ZSK. Recursively, the *com.* zone submits its DS record for inclusion to the root. It is the responsibility of the parent zone to validate the correctness of the DS records before inclusion. Algorithm 4 presents the setup procedure for each DNS zone to enable DNSSEC successfully.

Constructing DNSSEC Proof Chains

A recursive resolver $\mathcal{R}$ communicates with nameservers S_{root}, S_{tld}, and S_d to resolve a query for a FQDN d sent as a message Q_d from a client C. During its communications, the resolver obtains the required DNS public keys associated with the zone and key signing keys (ZSK and KSK), as well as a signature σ covering any other required RRs from each nameserver. This results in the resolver collecting various DNSKEYs – $(ZSK_{\text{root}}, KSK_{\text{root}}, \sigma_{\text{root}}^{\text{DNSKEY}})$, $(ZSK_{\text{tld}}, KSK_{\text{tld}}, \sigma_{\text{tld}}^{\text{DNSKEY}})$, and $(ZSK_{\text{d}}, KSK_{\text{d}}, \sigma_{\text{d}}^{\text{DNSKEY}})$. Similarly, the resolver also retrieves signed delegation signer (DS) records from the parent nameservers – $(\text{DS}_{\text{tld}}, \sigma_{\text{root}}^{\text{DS}})$ from the root nameservers, and $(\text{DS}_d, \sigma_{\text{tld}}^{\text{DS}})$ from the TLD nameservers. The signed DS records are verified using the associated zone signing keys of the root and the TLD nameservers, and are compared to hashes of the key signing keys ($\mathbb{H}(KSK_{\text{tld}}) = \text{DS}_{\text{tld}}$, $\mathbb{H}(KSK_d) = \text{DS}_d$). The resolver obtains the answer $\mathcal{A}$ and the associated signatures $\sigma_{\text{d}}^{\mathcal{A}}$ for the client query.

Intuition Behind DNSSEC Proof Chains: At a high level, as shown in Figure 7.2, the *proof chain* is constructed as a series of Zones with each Zone containing information about a parent zone (if any), the DNS keys, DS records associated with the immediate child zone in the DNS hierarchy, and the cryptographic signatures to validate these records. Additionally, the same structure contains an optional leaf record indicating canonical names for alternate chains, or the answer associated with the resolved query. Our format is inspired by [201]; with changes to decrease complexity. Foremost among these changes is that we do not cherry pick data from *within* the DNS resource records for inclusion into our custom Zone data structures. To maintain easy interoperability, all the individual RRs within each Zone are serialized as described in existing IETF standards, and the Zone in turn being serialized in a similar manner — allowing existing compression mechanisms to continue being used addressing information redundancies within the serialized records [231, 232].

Detailed Construction of Proof Chains: For a given query Q_d, the recursive resolver traverses the DNS hierarchy to obtain the answer $\mathcal{A}$ associated with the query. These traversals result in the resolver obtaining the associated DNSKEYs, delegations (DS) to the child zone from the parent, and authoritative information with the nameservers which the resolver needs to communicate with to continue the traversal. We selectively store only the necessary cryptographic material such as DNSKEYs, DSes, and their associated signatures when additional nameservers need to be contacted by the resolver. The resolver creates a list of generic DNS Resource Records which include the DNSKEYs, DSes from the current traversal of the DNS hierarchy. At each traversal which indicates a further delegation, the resolver appends the cryptographic key material and associated signatures to the initial set of records. For the final traversal where the resolver communicates with the nameserver $\mathcal{S}_d$ associated with the query Q_d, the resolver appends the cryptographic key material in addition to the expected signed answer or includes a denial of existence proof.

While each individual DNS RR structure and associated signatures are included without any

modification, additional information sent as glue records, and in the authoritative segment of the DNS messages from the nameservers are discarded completely since they do not add value to prove the correctness of the overall result. For example, it is unnecessary to learn about the nameservers of the TLD contacted and the delegations presented to obtain the result for the domain d since the signed cryptographic keys and delegation signer records at each level indicate the authenticity of the communication with the required nameservers. This reduces the size of the longer DNS responses (typically glue records) which are obtained from root, or TLD nameservers.

This approach to construct the DNSSEC proof chain as a nested list of list of relevant DNS resource records provides the inherent advantages of serializing the records using an IETF standardized network wire format. The data present in the proof chain needs to be strictly ordered (from root to leaf) and prevents clients from making assumptions about the resolution order. Additionally, it allows clients and resolvers to negotiate *key-tags* allowing only partial chains to be returned from the resolver since portions of these chains such as the DNSKEYs, and DSes at the root and TLD levels would remain constant and cacheable due to large TTLs associated. Using existing message formats and sharing required DNS RRs allows for easier verification of DNS messages in standard libraries without needing to re-implement cryptographic validation logic – reducing potential security challenges.

The proof chain is constructed such that a client traversing the proof chain has all the required information in the current segment of the chain, which would be required for the next segment. For example, the traversal of a chain of size 3 would contain the DS record associated with the TLD in the first segment of the chain in addition to information to compare and validate the root records. The correctness of the DS record information can immediately be checked when traversing to the second element where hashing the KSK present in the DNSKEYs would result in the expected DS records. The remaining information such as DNSKEYs, and the answers are self-contained in the existing traversal and can be validated immediately. As a result, the client has

minimal additional memory overhead when traversing the chain – limited to the byte size of the delegation signer. By employing this design, the proof chain contains the information needed to verify the authenticity and integrity of the data within each segment of the chain, and carries over minimal memory overheads to validate the correctness of the information between consecutive pairs of chain segments.

Validating DNSSEC Proof Chains

In our scheme, the client would be charged with validating the serialized proof chain. The local client DNS stub resolver would be bootstrapped in a secure out-of-band process with the DNSSEC root trust anchors containing the hashes of the root DNS keys [166] and a signature covering the contents. These could be the IANA root trust anchors for the Internet or a mutually configured anchor to a private alternative public key infrastructure. The signature of the trust anchor itself would be validated with the key in a trusted certificate [9]. While the standard IANA signed DNSSEC trust anchors are used in our implementation, our scheme also permits alternate trust anchors to be configured and used if desired.

Intuition Behind Verifying a Proof Chain: A simplified version of our validation procedure is shown in Algorithm 5. The procedure accepts as input the query Q_d, the *proof chain* $\mathcal{Z}_p$ containing the individual zones $\mathcal{Z}_i$, and a verified trust anchor containing the hashes of the root DNS keys H_{root}. The procedure then traverses the list of zones $\mathcal{Z}_p$. For each zone $\mathcal{Z}_i$, we check the signatures covering the signing keys, and establish proof chain continuity by comparing and validating the DS records. The last zone associated with the domain $\mathcal{Z}_d$ in the proof chain contains the answer $\mathcal{A}$ or a denial of existence proof (NSEC/NSEC3) [26, 286] for the given client query and an associated signature $\sigma^{\mathcal{A}}_{\mathcal{Z}_d}$ which is validated accordingly – returning a boolean response indicating the validity of the serialization and in turn the resolution. The serialized proof chain is included in the *Additional* section of the DNS message as its own resource record (RR) type [232].

Algorithm 5 DNSSEC Proof Chain Validation, Abridged

1: **Input:** Target: Q_d, Proof chain: $\mathcal{Z}_p$, and H_{root}
2: **Output:** Proof chain validity (**true/false**)
3: $trusted\_keys \leftarrow \{(\cdot) \rightarrow \text{KSK}_{\mathcal{Z}_{\text{root}}}\}$
4: **for all** zone $\mathcal{Z}_i \in \mathcal{Z}_p$
5: **Assert**(Verify($\mathcal{Z}_i^{\text{DNSKEY}}$,$\sigma_{\mathcal{Z}_i}^{\text{DNSKEY}}$,LookUpKSK($\mathcal{Z}_i$)))
6: Append($\text{ZSK}_{\mathcal{Z}_i}$, $trusted\_keys[\mathcal{Z}_i]$)
7: **if** ExistsInZone(DS, $\mathcal{Z}_i$) **then**
8: DSValidity $\leftarrow$ Verify($\mathcal{Z}_i^{\text{DS}}$, $\sigma_{\mathcal{Z}_i}^{\text{DS}}$, LookUpZSK($\mathcal{Z}_i$))
9: HashEquality $\leftarrow$ **IsEqual**(($\mathbb{H}(\text{KSK}_{\mathcal{Z}_{i+1}})$, $\mathcal{Z}_i^{\text{DS}}$)
10: **Assert**(DSValidity $\wedge$ HashEquality)
11: Append($\text{KSK}_{\mathcal{Z}_{i+1}}$, $trusted\_keys[\mathcal{Z}_{i+1}]$)
12: **end if**
13: **if** ContainsLeaf($\mathcal{Z}_i$) $\wedge$ $Q_d \in \mathcal{Z}_i$ **then**
14: $\mathcal{Z}_d \leftarrow \mathcal{Z}_i$; $\mathcal{A} \leftarrow \mathcal{Z}_i^{\text{Leaf}}$;
15: **return** Verify($\mathcal{A}$, $\sigma_{\mathcal{Z}_d}^{\mathcal{A}}$, LookupZSK($\mathcal{Z}_d$))
16: **end if**
17: **end for**
18: **return false**

Detailed Validation of Proof Chains: A client obtaining a DNS response from a DNSSEC serializing resolver would observe the proof chain in the *Additional* section of the response message. The client stub resolver, parses and de-serializes this section containing an RR type to obtain the nested list of lists representing the proof chain. The proof chain is validated from the root to the leaf and begins by mapping the root zone to the keys obtained as a part of the trust anchor configuration for the client. For each list containing the zone specific information in the proof chain, the cryptographic key material such as DNSKEYs are validated by using the standard Verify methods provided by public key cryptographic libraries [181]. The Verify method takes the message data to be verified such as the DNSKEYs, DS records, the signatures presented as RRSIG records (σ) and the public part of the associated signing key. To verify the correctness and validity of the DNSKEYs in the current segment of the proof chain being traversed, the KSK of the zone is used as shown in line 5 of Algorithm 5. The associated ZSK in the DNSKEYs are added to the list

of trusted keys.

The client then identifies the existence of delegation signer records (DS) which is enabled by the method ExistsInZone(). The signature associated with the DS record for the child zone is verified using the ZSK of the current parent zone. The DS record information contains information about the cryptographic hash function to apply on the KSK of the subsequent zone record. The KSK record in the following segment of the proof chain being traversed is hashed with the algorithm specified in the DS record and compared. The process of verifying the DNSKEYs, DS records, and preparing the list of trusted keys (used for caching) is iteratively performed on all segments of the proof chain. The ContainsLeaf method identifies the existence of the answer record or a proof of non-existence (NSEC/NSEC3) associated with the client query in the current zone. This check would not yield a DNS RR at any stage of proof chain validation except the last record. The integrity, and authenticity of the answer $\mathcal{A}$ included in the leaf is validated and accepted by the client as a valid answer associated with the requested query. Through this process, the client successfully validates the authenticity of the resolver response without implicitly trusting the resolutions provided by it. The resolver on the other hand is successfully able to prove its ability to operate honestly and correctly to the clients leveraging its services.

Implementation Specifics

To reduce the scope of changes and maintain backwards compatibility, our design requires only modest changes to client queries. We propose including a new EDNS0 bit indicating that a client is requesting the serialized proof. We refer to this serialized proof bit as SP used by the recursive resolver. To receive the serialized proof, this bit must be included along with the DO bit. However, this could be achieved with no adverse impacts by repurposing the intent of the DO bit. While the proposed design requires client side software updates, we believe the implementation for this is relatively straightforward since the validation of the proof chain relies on already existing library

functions used for validating DNSSEC responses.

We implemented a validating recursive DNS resolver in Golang supporting various protocols such as Do53 using UDP and TCP, DNS over HTTPS (DoH), and Oblivious DNS over HTTPS (ODoH) protocols. We re-used a popular and widely used DNS library ($\mathrm{miekg/dns}$) as the core for building the recursive resolver [133]. DNSSEC enabled client queries to the resolver resulted in the resolver returning the proof chain with the response. We modify the DNS library to introduce a new DNS RR type associated with an authenticated proof chain and implement the corresponding $\mathrm{Pack}()$ and $\mathrm{Unpack}()$ methods responsible for converting the RR into a standardized wire format. Since we want to enable clients to identify the correctness of the resolver, despite local validation, the resolver is configured to continue providing responses to queries which fail DNSSEC validation. We experiment with various incorrect and well known experimental DNSSEC zones, all of which are correctly validated by the client.

Similar to recursive resolvers deployed in the real world, the resolver implementation is configured to maintain a cache which is empty when run for the first time. The caches are configured to immediately cache DNS RR information about the root and the various children TLD levels. A client query received at the resolver results in the resolver greedily optimizing and identifying all necessary DNS sub-queries which need to be made. These sub-queries such as obtaining the DNSKEY, DS, records are resolved in parallel whenever possible. The responses from the various cache lookups as a part of the resolution, and the necessary communications with the authoritative nameservers are assembled in a strict traversal order. The set of these assembled DNS RRs are converted into the new DNS RR type for proof chain that we introduce and included in the *additional* section of the DNS response.

We implement a custom Golang based client with an interface similar to the popular DNS lookup utility dig. In addition to including the support for validating the DNSSEC proof chain, we include support for the client to behave as a recursive client stub using TCP and UDP transports,

and integrate benchmarking suites for various DNS protocols. The client is bootstrapped with the IANA root zone trust anchor.

7.1.5 Evaluation and Results

Experimental Setup

We deploy an instance of the recursive resolver on Google Cloud using the $\mathrm{c2\text{-}standard\text{-}16}$ machine with 16 CPU cores and 64 GB of memory in the US West 1B data center located in Oregon, United States. For benchmarking the resolver performance with ODoH, and to emulate the distribution between different organizations, we deploy a proxy on Amazon Web Services EC2 in the US West 1 data center location in Northern California, United States. We run our benchmarks using a dedicated client machine located in a university network connected over a WLAN maintaining approximately 100 concurrent requests in flight at any time to our resolver. We believe this chosen experimental setup models a typical Internet user connecting to a DNS resolver through their Internet service provider, and accounts for congestion and concurrent use of the network by various clients.

For benchmarking, before running each measurement, we clear and re-initialize the recursive resolver cache to model a cold start. We randomly sample 10,000 domain names from the Tranco top million dataset which we provide to the client to resolve their IPv4 address [205]. We measure the overhead of our scheme over both the TCP and UDP transport protocols, as well as the baseline performance for DNS queries without including the proof chain. Prior measurements indicate the importance of connection reuse for DNS protocols [80, 310, 359]. To provide the benefits to the client and mimic existing client behavior, our implementation of the client reuses existing network connections whenever available. For example, the TCP connections are reused in protocols relying on the underlying TCP transport to avoid unnecessary TCP handshakes and the TLS connections are reused during DoH, and ODoH benchmarks to avoid unnecessary TCP and TLS handshakes.

For the ODoH benchmark, our recursive resolver implementation acts as a co-located resolver and ODoH gateway, with the proxy being hosted on a different public cloud network. During the measurements with DNS over HTTPS over Tor (DoHoT), we configure the ability of the client to communicate over Tor by tunneling the DNS requests over Tor using SOCKS5. Additionally the Tor nodes are configured to choose the Entry, Middle, and Exit nodes within the United States and are configured to create their own optimal routing circuits.

Results

It is widely considered that the usage of DNSSEC slows down DNS resolutions because of the increased packet sizes obtained in the responses, the number of queries and responses needed between the client and the recursive resolver, and the added overhead of verifying the returned signatures against their signing keys [211]. Our measurements show that requesting the serialized proof chains increases the query size (due to additional bits being set) and the response answer size (due to the inclusion of the *entire* proof chain) when requesting DNSSEC enabled responses. Clients configured to recursively traverse the DNS hierarchy without using a recursive resolver are configured to be able to request DNSSEC responses. The experiments requesting the DNSSEC responses are compared to those where the recursive resolver returns a proof chain, while those that do not return the proof incur the networking overheads of communicating with authoritative nameservers.

Cryptographic Compute Overheads: The client incurs a negligible compute verification time overhead to verify the *entire* proof chain causes. Through our micro-benchmarks, we observe compute overheads due to validating the proof chain incurring 0.338 ms median, and a 1.59 ms p99 overhead accounting for $\leq$0.5% of the average query resolution latency.

Protocol	Proof?	Network Query Size		Network Answer Size		*p50 Proof Overhead*	*p99 Proof Overhead*	DNS Response Time (ms)			
		p50	*p99*	*p50*	*p99*			*Mean*	*p50*	*p75*	*p99*
Do53 using UDP	✓	42 B	76 B	2172 B	4109 B	31.94x	2.37x	312.52	172.54	410.17	2001.46
	✗	31 B	65 B	68 B	1731 B			325.67	172.80	401.70	2001.72
Do53 using TCP	✓	42 B	73 B	2178 B	8175 B	31.11x	4.85x	347.97	192.29	444.39	2018.98
	✗	31 B	72 B	70 B	1684 B			336.03	191.42	437.21	2018.73
DNS over HTTPS (DoH)	✓	42 B	90 B	2180 B	7536 B	32.05x	4.44x	361.43	173.44	422.46	3009.85
	✗	31 B	74 B	68 B	1697 B			342.99	171.55	413.87	2508.12
Oblivious DoH (ODoH)	✓	131 B	157 B	2225 B	7207 B	20.04x	4.63x	363.51	196.73	448.00	2553.86
	✗	120 B	170 B	111 B	1555 B			323.35	194.26	421.43	1711.65
DoH over Tor (DoHoT)	✓	42 B	83 B	2180 B	7315 B	31.14x	5.67x	5879.24	5421.34	6970.61	18262.70
	✗	31 B	74 B	70 B	1289 B			2038.51	1576.31	1836.07	18608.41
Recursive Client Do53 with UDP	✓	221 B	386 B	625 B	2883 B	1.00x	1.01x	404.55	229.00	583.12	2327.99
	✗	221 B	289 B	623 B	2854 B			401.61	221.51	580.42	2512.07
Recursive Client Do53 with TCP	✓	221 B	380 B	3242 B	8259 B	1.00x	0.97x	505.95	251.68	731.68	3219.75
	✗	221 B	386 B	3240 B	8535 B			586.35	350.42	809.39	3618.57

Table 7.1: Proof-chain inclusion overheads for on-wire network query, answer bytes and response timings by DNS protocol.

Impact on DNS Response Sizes: Using Do53 UDP as the baseline since it is the most widely used DNS protocol, our measurements indicate a P99 increase in answer size due to the inclusion of the serialized proof chain is ≈2.3x (4109 bytes) compared to when clients do not request any serialized proof chain (1731 bytes). The response sizes due to the proof chain is 1.44x larger (4109 bytes) than UDP based DNS clients recursively traversing the DNS hierarchy to validate necessary records which incur a P99 overhead of 2883 bytes. However, at the median, the difference due to the proof chain inclusion is stark – indicating a 31x increase (2172 bytes) in the median response sizes when Do53 UDP clients request an answer from a validating recursive resolver. The client obtaining these responses implicitly trust the DNS resolver and cannot validate the proof chain. To truly compare the difference, we compare the Do53 UDP client retrieving a proof chain to a recursive UDP Do53 client performing all required actions by itself. The median response size when including the proof chain from a validating resolver is 2172 bytes which is 3.4x the networking overhead observed by a recursive Do53 UDP client (625 bytes). However, despite the larger sizes, returning a serialized proof from a resolver using Do53-TCP indicates a 50% improvement (2178 B) compared to median network overheads from recursive TCP clients (3,242 bytes) — due to additional TCP/IP and handshake overheads. For each protocol in our measurements, we present

the network overhead to make the query, and the response overheads due to the inclusion of the proof chain in Table 7.1.

To understand the impact of the proof chains in DNS responses, we use Chromium and configure the system DNS stub resolver to use the deployed recursive resolver. We choose 10000 random websites from the same Tranco list used for prior measurements and instruct the browser to perform a full page load and measure the largest contentful paint (LCP) metric involving all necessary subresources being resolved. The same websites and the order of page loads are maintained for both measurements with caching disabled. One set of experiments return the serialized proof chains, while another does not and all experimental runs use the Do53 UDP based DNS protocol. We observe that the median LCP page load time is 1629.0 ms when using the serializing resolver and shows a 1.6% median improvement over using a resolver not returning the proof chain whose median LCP page load time is 1655.5 ms. As shown in Figure 7.4, when using a resolver without the proof chain in the response, the average web page loads 1% faster (11ms) than when a proof chain is returned – indicating minimal impact on page load time experiences , or comparable results to web page load times due to the inclusion of the proof.

Impact on DNS Response and Page Load Time Latency: As shown in Table 7.1, our measurements indicate that despite the increase in the response sizes, the latency for responses between the client and the resolver remain comparable to today's conditions where the client does not request the information from a resolver with the DO bit set. The results indicate that the mean response time for TCP-based plaintext DNS over port 53 from a validating resolver due to the inclusion of a the serialized proof chain increases by 3.55% from 336.03 ms to 347.97 ms. Similarly the usage of UDP-based DNS over port 53 resulted in a 4% *improvement* in mean response times. While the observations imply improvements, many factors outside the control of the experiment could have influenced these results such as the random choice of queries, and the routing or re-assembly of potentially fragmented IP packets. Therefore we emphatically refrain from considering these as

improvements but rather as indicators that the inclusion of serialization in the response is strictly *no-worse* for latency. We compare these results to the latency incurred by clients performing the recursive resolutions themselves as shown in Figure 7.5 and observe 22% (404.55ms ⟶ 312.52ms) and 31% (505.95ms ⟶ 347.97ms) average improvements when using a recursive resolver (green and orange lines, v/s purple and red). The inclusion of the proof chain has no adverse latency impacts indicating the practicality of our solution.

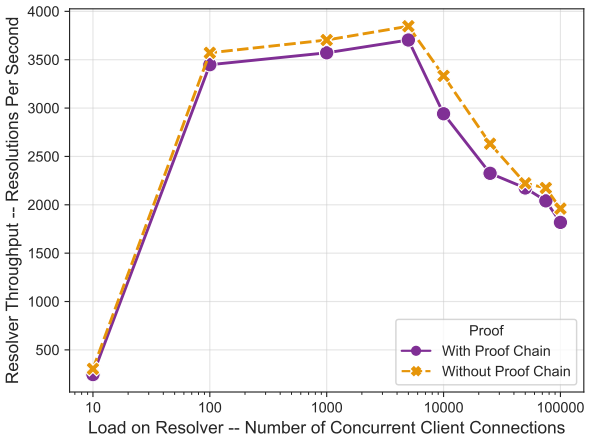


Figure 7.3: Microbenchmarks showing the impact of request load and proof chain inclusion on DNS resolver throughput.

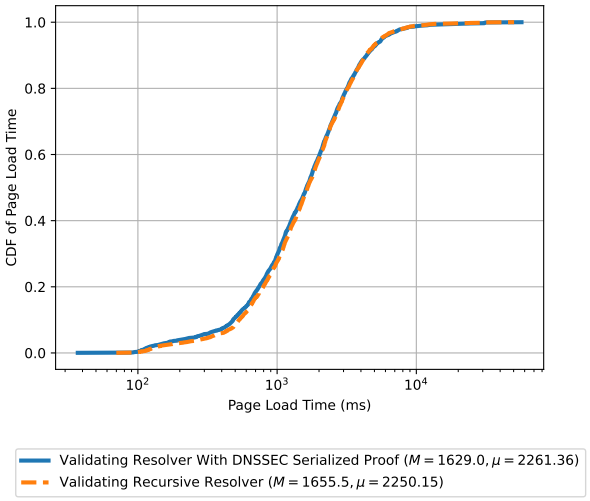


Figure 7.4: Impact of Proof Chain Inclusion on Page Load Time

Impact on Recursive Resolver Throughput: While our evaluation of the client side indicates no noticeable performance regressions, it is important to understand the implications for the recursive resolver. We perform micro-benchmarks by hosting the recursive resolver on a constrained VM instance with 1 CPU and 2GB of memory running 64 bit Ubuntu Server 22.04. Figure 7.3 shows the results of stress testing the resolver instance using a set of client VM instances configured on the same subnet. The client VMs are configured to consistently perform DNS requests at a specified rate to meeting the experimental condition for the active load on the resolver. Combined,

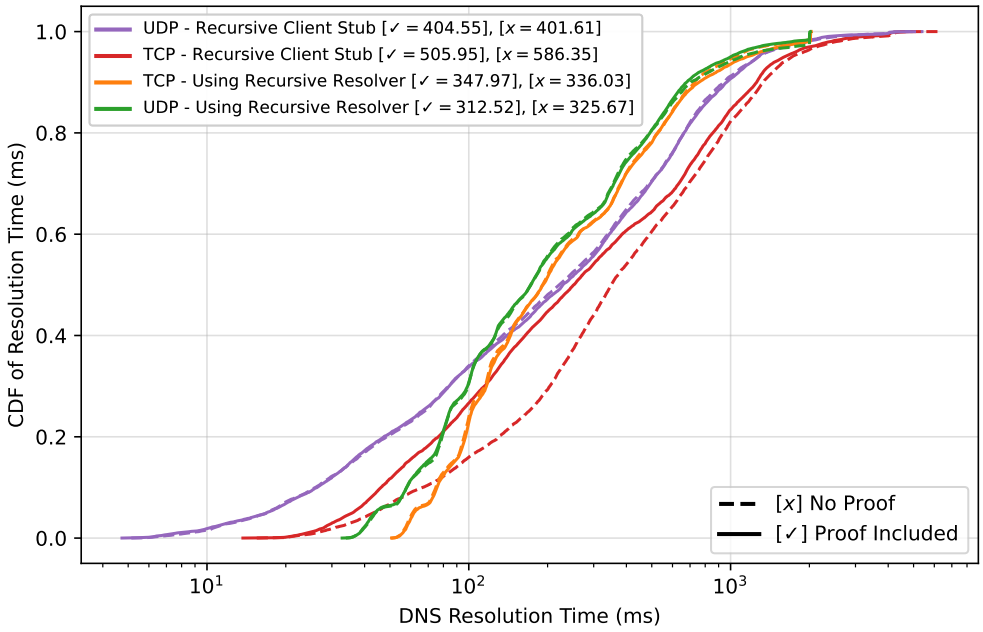


Figure 7.5: DNS response latency for clients performing recursive resolutions, compared to using a recursive resolver via UDP and TCP transports. (Average latency included in legend)

the clients attempt 100K DNS resolutions and the throughput resolutions per second is calculated.

As shown in Figure 7.3, we observe a slight reduction in the throughput of the resolver when clients request proof chains (solid purple line), compared to when they do not request the proof chains (dashed orange line). At low loads on the resolver, the resolver is performing lesser work resulting in lower throughput since the clients are not issuing enough DNS requests to stress the resolver's capabilities. At its peak, the resolver successfully resolves 3846 DNS requests per second when proof chains are not requested. The throughput reduces by 3.69% to 3704 DNS requests resolved per second. At higher server loads, the throughput of the resolver reduces to roughly half its peak throughput capacity with 1818 resolutions per second with inclusion of the proof chain and 1960 resolutions per second without the proof chain being included in the DNS response. While our proposed method increases the compute overheads incurred by the recursive resolver, it is important to understand that these services when exposed to the public

as open resolvers operated by Google, Cloudflare, NextDNS are designed to be highly scalable with automatic infrastructure scaling and load balancing. The overheads we introduce while small in nature (≈3% drop in throughput) provide significant advantages by allowing resolver operators to prove their authenticity and trustworthiness to their clients. While we do not like to downplay the implications of a 3% reduction in throughput when operating at scale, we believe that existing infrastructure over-provisioning practices (such as the ones employed by our partner CDN) and improved caching mechanisms can absorb this performance penalty and continue to provide highly available and reliable DNS resolutions.

We believe, our results strongly indicate the immediate viability of our proposed solution, provides significant advantages by reducing implicit trust placed by the client in the resolver services and has unnoticeable to no performance regressions experienced by the client.

7.1.6 DNSSEC Validation Prevents Client Privacy Leakage

The importance of DNSSEC in today's Internet has largely been downplayed due to the presence of the web TLS certificates which are successful at identifying threats such as navigating to resources with invalid or untrusted TLS certificates. While the guarantees provided by certificates and the browsers using the CA root trust stores are effective at preventing users from navigating to malicious web resources, the underlying TCP + TLS handshakes to a malicious server resolved via DNS reveal the existence of a client that attempted to connect to it – leaking the privacy of the client and information such as the client IP address which can be used in combination with GeoIP datasets to identify the city, country or ISP of the client.

DNSSEC validation and the usage of a recursive resolver, reveals the identity of the resolver, an entity used by many others within the network to the nameservers of the resource being resolved – preserving the privacy of the individual client. Responses from the nameservers which fail DNSSEC validation at a validating resolver or the client stub due to the usage of the proof chain

would actively prevent the client from attempting a connection to the web resource protecting their privacy from potential attackers. Coupled with the guarantees provided by the TLS certificates and the web PKI ecosystem, this opens up opportunities for browsers to implement explainable error messages informing the users why the browser cannot complete the navigation to the requested resource – a clear improvement over displaying an unreachable NXDOMAIN webpage (Chrome), or a generic connectivity error (Firefox).

7.1.7 Distributed Denial of Service (DDoS) Implications

The existence of open DNS resolvers such as public recursive resolvers create challenges for authoritative name servers due to the ability for attackers to amplify the traffic targeting a specific set of nameservers and overwhelming them. The deployment of the proof chain inclusion however does not worsen existing amplification attack risks. This is because the communication between the recursive resolvers and the authoritative nameservers are unmodified. The responsibility for the proof chain creation rests only on the recursive resolvers who experience a slight increase in computational overheads affecting their overall throughput. However, increased message sizes and computational overheads increase the denial of service risks on recursive resolvers. It is therefore important for resolvers providing proof chains to implement effective caching strategies preventing frequent communication with authoritative nameservers, employ over-provisioning as an operational strategy to absorb attack traffic, deploy network firewalls implementing rate control mechanisms, encourage various ISPs to implement source address validation effectively *sinkholing* spoofed network traffic that could become responsible for DDoS attacks [37, 308, 320].

Such operational safeguards are implemented by large public recursive DNS resolver operators such as Cloudflare, Akamai, NextDNS, and Google etc..,. Additionally, we believe the DDoS risks due to increased message sizes are neither improved nor worsened by the adoption of the inclusion of proof chains as proposed in this work. We believe that the risks outweigh any potential

disadvantages due to improved security guarantees providing authenticity and integrity to the DNS responses. Confidentiality and privacy guarantees could also be obtained by using existing secure transport protocols which require no changes with our proposal. Since most resolver infrastructure is situated at network boundaries, the increased size of responses are localized to the traffic within the network of an ISP, or an organizational network.

7.1.8 Introducing DNS Auditability Capabilities

In the Web PKI ecosystem, Certificate Transparency (CT) was adopted to promote visibility and defend against rogue CAs distributing incorrectly signed certificates [202]. The hierarchical nature of DNS, its rapid churn in content, and its distributed nature makes solutions like CT a non-starter. Our proposed proof chains however, offer assurance that the resolution received was correct and verifiable — at least at one moment in time. This offers timestamped auditability to clients and resolvers who can later prove their actions based on the DNS resolution as justified.

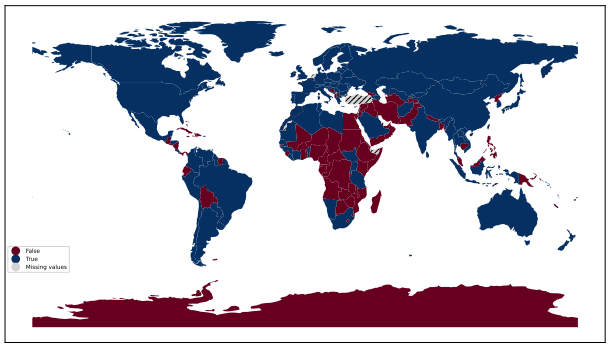
Such ability to prove DNS interactions is of high value to existing certificate authorities. Currently it is the responsibility of the certificate issuing authority to validate the hostname for which the certificate is being requested through a variety of standardized ACME protocol challenges such as HTTP-O1, DNS-01, or TLS-ALPN-01 [47]. Using DNS based challenges to prove hostname ownership to the CA is a common practice by administrators requesting TLS certificates. As a result, CAs use validating recursive resolvers and multi-vantage point probes to safely confirm the ownership of the hostname and rule out potential DNS based hijacking attempts or BGP route hijacks. Using proof chains, the CAs could issue certificates and include the serialized proof chain used during their challenges to demonstrate the correctness of their domain validation – thus enabling mechanisms to understand how different hostnames were validated prior to certificate issuance. Further, an included proof chain in the certificate extensions such as the one standardized by Dukhovni *et al.* [115] could support bindings between the DNS and TLS

ecosystems which are missing today, and could be coupled with existing efforts in improving the adoption of DANE as an authentication mechanism using the DNS ecosystem as an alternate public key infrastructure [44, 114]. A binding could also be leveraged by web servers to provide clients with verifiable resolutions for sub-resources in HTTP frame headers albeit this would require browser changes [251, 314, 331]. Some Web3 infrastructures mandate the usage of DNSSEC. IPFS in particular has experimented in the past with schemes to pass domain-to-content mappings through an untrusted IPFS HTTP gateway [224]. While their solution is only applicable to IPFS, our serialization mechanism is generalizable – opening up opportunities to expand the possibilities of a secure Internet, one with the potential to enable new applications and one that allows DNS infrastructure to interoperate securely with the next generation of web technologies.

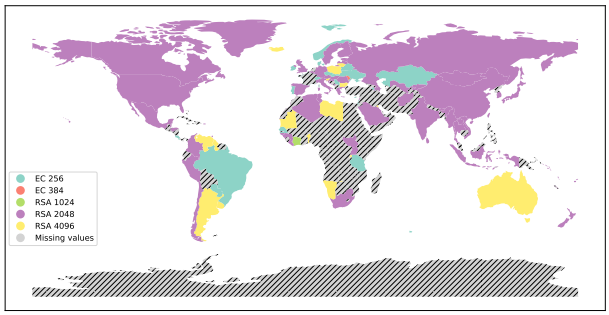
7.1.9 Current State of DNSSEC Adoption

DNS due to its hierarchical nature requires the adoption of DNSSEC at each zone in the hierarchy. If the parent zone (eg. TLD) does not use DNSSEC, then its children zones cannot enable and take advantages of DNSSEC. Today, there are still many country code TLDs that do not support DNSSEC making all its children zone unable to take advantages of security properties that DNSSEC enables. Our measurements of the ccTLDs indicate that of the 316 available ccTLDs in active use, 145 (45.88%) are DNSSEC enabled while the remaining 171 do not yet use DNSSEC. Newer zones being registered are mandated to use DNSSEC. Figure 7.6a reflects possible systemic inequities in the adoption of DNSSEC in global country TLDs with those colored blue adopting DNSSEC while the red ones have not yet adopted DNSSEC.

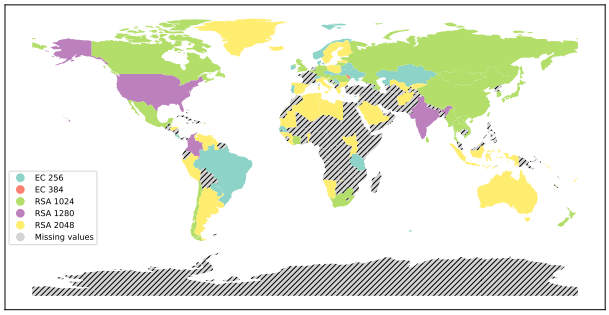
Grouping the adoption of DNSSEC by continent, we see DNSSEC adoption in 34% in the countries in Africa, 50% in North America, 58% in Asia, 70% in South America, 75% in Oceania, and 90% in Europe. In addition to measuring the adoption of DNSSEC among the TLDs, We also analyze the public key algorithms, cryptographic key lengths, and the hashing mechanisms used



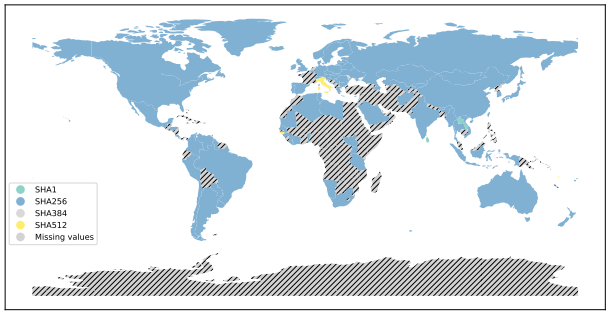
(a) DNSSEC support at Country TLD Zone Level



(b) Type of DNSKEY Key Signing Key (KSK)



(c) Type of DNSKEY Zone Signing Key (ZSK)



(d) Type of Hash function Used

Figure 7.6: Global DNSSEC Adoption, Key Types and Hash Algorithm Requirements

by different ccTLDs. We present our findings for the type of key algorithms and their lengths used in Figure 7.6d. Specifically Figure 7.6b indicates the public key algorithm type and the size of the key used as the Key Signing Key (KSK) to protect the integrity of the DNSKEYs, Figure 7.6c presents the key type and lengths for the Zone Signing Keys (ZSK), while the Figure 7.6d presents the type of hash function used during signing with the secure SHA256 being the most dominant. While keys with higher bit strength security are used as the KSKs, the ZSKs are typically of a lower security level – possibly due to the need for frequent signing and key rotation practices.

7.1.10 Conclusion

DNS resolvers are critical components of the Internet infrastructure and are implicitly trusted by their users to give correct responses. Leveraging the fact that a validating recursive resolver already incurs communication overheads during the traversal of the DNS hierarchy to resolve a client query, our proposal makes a simple but highly effective change by serializing the necessary DNS records and associated cryptographic signatures to prove the correctness of the resolution. Through extensive experiments we show that the proposed solution can be deployed immediately by large resolver operators, has no adverse impact for the clients whose latency and page load time measurements are unaffected, introduces a minor computational overhead slightly reducing the throughput of the resolver, and reduces the implicit trust placed in the resolvers – making them transparent and auditable. These improvements significantly reduce the need for users on the Internet to trust their DNS resolver and is complementary to securing DNS communications.

7.2 Verifiable and E2EE Cellular Interactions

As Internet applications become increasingly reliant on cellular identities, the ability to gain access to secure applications on the Internet such as messaging depends on the benevolence of the cellular network operator in not filtering the incoming one time passwords. While the decentralized authentication system dAuth presented in §4 allows users to seek services from other operators for cellular services, the plaintext nature of Short Messaging Service (SMS), despite the decades of innovation in cellular networks, poses a significant privacy challenge.

Ideally, the network operator responsible for message delivery to the intended user must not be able to read and analyze sensitive messages such as password resets, medical reminders, banking transactions, and one time passwords. Over a trillion messages like these are sent every year from businesses to their customers and they're a compelling mechanism for multi-factor authentication for Internet based applications because of the explicit hardware identity guarantees and their

ability to delegate identity challenges to the cellular networks. As the boundaries between cellular networks and Internet service providers enabling users access to the Internet blurs, it is important for user privacy to enable secure cellular communications.

Interesting opportunities present due to regulatory mandates such as the European Union's interoperability requirement for E2EE messaging providers enforced by the Digital Markets Act [208]. These interoperability mandates result in a requirement for fundamental change to the centralized architecture of popular E2EE messaging platforms such as WhatsApp, or Signal, among others, and various architectures have been proposed to address this challenge – from client only changes to the introduction of trusted proxy infrastructure. However, such interoperability has long existed in cellular communications deployed around the world and present an alternative position supporting the migration of today's centralized E2EE messaging infrastructure to cellular infrastructure. Such migrations reduce the trust placed in a centralized communications service provider and split trust by enabling a decentralized and distributed ecosystem – similar to the global Internet network.

7.2.1 Reducing Trust in Cellular Third Parties/Intermediaries

Bootstrapping a user onto a secure application like Signal or WhatsApp relies on the usage of user's phone number issued by the carrier network as a proxy for identity and uses the SMS based one time password to complete the verification to prevent abuse of the registration system by adversaries. The services provided by the cellular networks, like SMS, play a crucial role in ensuring various Internet applications work correctly and prevents denial of service attacks by providing sybil tolerance – preventing malicious users from creating fake accounts. This has given rise an entire ecosystem of hidden entities which developers of Internet applications interact with that act as gateways between the Internet and global cellular networks, providing authentication and messaging services. We argue that it is *unnecessary* for these third party providers operating

application gateways, or their position between core networks for interoperability, to learn about the contents of the messages. Yet, they are trusted with the same information. With recent attacks such as the ones on Syniverse, resulting in over a trillion messages being leaked, it's immediately clear that addressing these problems is long due.

7.2.2 The Identity Crisis – Equivocation Risks in Centralized E2EE Systems

Internet based end-to-end encrypted communication systems like WhatsApp or Signal maintain identity systems mapping authenticated users via their phone number or associated hashed identifiers to their registered public keys. A user device signing up for the service generates a cryptographic key pair using the TEE on the device and securely transmits the contents to the communication service provider's servers. A key challenge, however, in enabling E2EE is the secure maintenance and retrieval of the public keys corresponding to the recipient on demand. The application is only secure if the participants in the process obtain the required keys and verify the integrity of the keys. Users today trust centralized applications such as WhatsApp or Signal to enable this by maintaining and requiring users to trust their key directory services and the responses that are provided to the application when communicating between two parties. While these applications provide ways in which two participants of a communication can validate the shared secret by validating a generated sequence of characters out of band in another trusted communication channel or by meeting in the real world, this is difficult for most users to do especially, when users rotate keys periodically or switch devices and user accounts.

The centralized nature of these hosted key directories could legally be compelled or compromised, resulting in services performing equivocation attacks and successfully intercepting and decrypting the intended communications [34, 347]. The following sequence of events between the server $\mathcal{S}$ containing the key directories, the sender C_s and the intended recipient C_r demonstrate a successful equivocation attack.

Setup($\mathcal{S}$):

$\text{SK}, \text{PK} \leftarrow \text{KeyGen}()$

$T \leftarrow [\,]$ ▷ Key Directory

$T[C_s] = \text{PK}_{C_s}$

$T[C_r] = \text{PK}_{C_r}$

func **getKey(C):**

return $T[C]$ **if** $C \neq C_r \wedge C \in T$ **else** PK

Messaging Workflow:

$C_s \rightarrow \mathcal{S}$: $\text{getKey}(C_r)$

$\mathcal{S} \rightarrow C_s$: PK ▷ *Equivocated Key*

$C_s \rightarrow C_r$ via $\mathcal{S}$

$C_s \rightarrow \mathcal{S}$: $M = \text{Enc}(m, \text{PK})$

$\mathcal{S} \rightarrow C_r$: $\text{Enc}(\textbf{Dec}(\boldsymbol{M}, \textbf{SK}), T[C_r])$

The messaging workflow indicates the equivocation attack where the server returns an indistinguishable public key whose private key it is fully aware of. The sending client has no way to verify the keys without meeting the recipient and validating the keys through an out-of-bands mechanism. The sending client encrypts its intended message m using the key provided and sends an encrypted message M to the recipient. The malicious server can then recover the plaintext message through $\text{Dec}(\text{Enc}(m, \text{PK}), \text{SK})$ highlighted in **bold**, which could be forwarded to the recipient C_r by performing the encryption on the tampered data if needed making it appear like the message is legitimate and sender originating. While approaches like CONIKS [226] and SEEMless [75] enable addressing the challenges of equivocation through Verifiable Key Directories (VKDs), I posit in this thesis that the reduction of trust in cellular carrier networks and the rise of smaller carrier networks due to the democratization of cellular networks opens up opportunities to maintain distributed VKDs without relying on a centralized key directory – enabling E2EE in cellular networks in addition to interoperability between E2EE communication providers and their auditability.

7.2.3 A Challenging Migration to Public Key Cryptography in Cellular Cores

The evolution of next generation cellular standards such as 6G, and the usage of TEE hardware in mobile devices pose opportunities for cellular networks to adopt public key cryptography to maintain user identities for authentication to the cellular networks – migrating away from the symmetric key based identities in existing 5G or older generation networks. Cellular networks maintain user identity by mapping the user to their associated symmetric key present in the SIM card ($U \leftrightarrow K_u$). With the evolution of eSIM infrastructure the issuance of SIM identifiers can be upgraded to public keys where a certified public key associated with the user is maintained by the core network. However, this migration to public keys requires hardware capabilities and updates to client side software and would require extensive standardization effort before user devices are manufactured.

Establishing and Connecting With Existing Public Key Infrastructure

The migration to public key cryptography in cellular core networks enables opportunities by cellular network providers and intermediaries to securely share verifiable cryptographic information with Internet applications. However, this poses a challenge as it would require the cellular network operators to enable publicly accessible PKI infrastructure to enable auditability. Today, unlike the web PKI infrastructure where multiple root CAs are able to issue certificates, all device certifications in the eSIM based SIM provisioning rely on profiles that are cryptographically signed by the GSMA key. Today, only 3 organizations (Cybertrust acquired and operated by Verizon, Digicert, and SEAL SQ, by WiseKey) provide the PKI on behalf of GSMA for remote SIM card provisioning globally. 2 out of the three (Cybertrust and Digicert) issuers are registered in the United States, while one (WiseKey) is registered in Switzerland. Similar to the previous measurements of government infrastructure, the choice of root certificate issuers within 2 countries for global cellular eSIM issuance leaves operators across the world vulnerable to MITM attacks due to

compelled certificate creation attacks. Unlike the web PKI ecosystem, the cellular PKI ecosystem does not yet have a CT log mechanism to establish transparency making it extremely difficult to observe misbehavior due to malice or compromise.

Arguably, it is valuable for any cellular PKI infrastructure to inter-operate with the web PKI that is relied upon for validating the host and establishing TLS handshakes. As regular software updates modify the trust anchors, these updates would make their way into browsers and various operating systems. This would enable the existing system of monitors and auditing to continue with minimal changes if necessary. However, while out of scope for this thesis, the additional burden on the auditors due to these changes must be studied closely.

Establishing Namespaces and Preventing User Identifier Collisions

A key challenge when directories (key-value stores) are sharded is to identify which shard holding the required information. In addition to human understandable country code prefixes for phone numbers, cellular networks, due to the standardization by ITU, allocate unique prefixes to both the IMSI SIM identifiers and the operators across the world. These unique allocations serve as *lookup directories* and split the global subscriber database. These practices reduce the collisions associated with user identifiers such as phone numbers across providers and enable efficient lookups since the challenges with *routing* the communications between a sender and a recipient is already currently addressed by cellular networks through the mobile switching center (MSC) and various signaling services orchestrated through SS7. Similar to BGP routing on the Internet, cellular networks advertise the routes to other networks to their peers, these could be by country prefixes, or longer country code, operator, and region based prefixes.

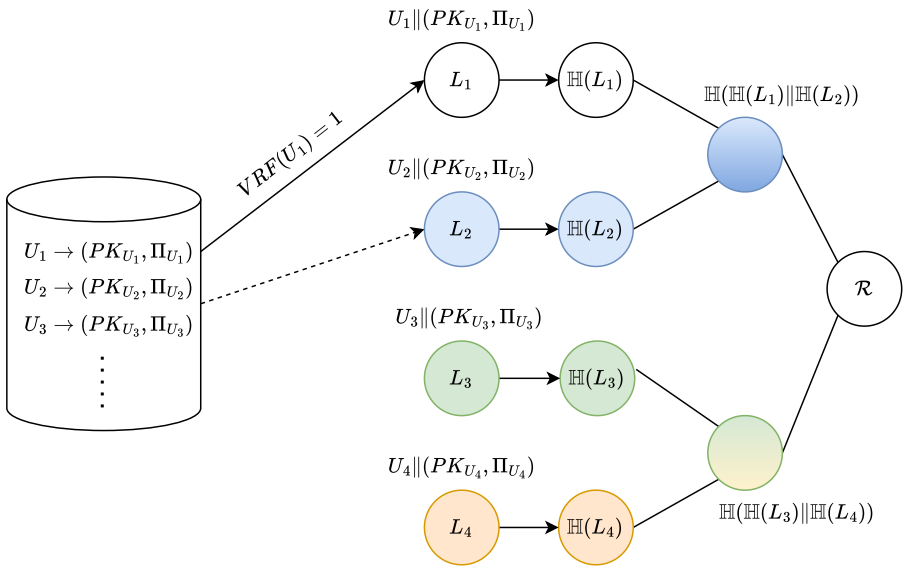


Figure 7.7: Mapping cellular subscriber key directories into a Merkle tree

7.2.4 Integrating Verifiable Key Directories into Cellular Core Networks

Designing Verifiable Key Directory Services

The establishment of public key directories for users does not yet address the challenges associated with equivocation attacks as presented in §7.2.2. The integrity of the key directories can be established by building auditable data structures, such as Merkle trees, where identity information such as their phone number associated with each subscriber is included as a label and the public key of the user as the value creating the label-value pair. These label-value pairs form the data-leaves of the Merkle tree and are hashed using a collision resistant hash function forming the first level of the tree. Pairs of leaves are concatenated and hashed together recursively until the root.

A Verifiable Random Function (VRF) is a well known function that generates a pseudorandom output allowing only the holder of a secret key to compute the result, while enabling a verifier knowing the public key to verify the correctness of the computation [135, 228]. These cryptographic

functions have been standardized and used in various real world systems [62, 134, 216, 294]. These functions, unlike one way cryptographic hash functions, are designed to not reveal information about the presence of other closely related information. The design of VKDs in cellular systems could use a simpler form of a VRF known as a verifiable unpredictable function (VUF) that allows mapping specific user information such as a phone number to a unique index [228]. Each leaf in the Merkle tree as a result contains the computed value from the VRF function in addition to the label-value pairs.

Each subscriber i registers their public key PK_{U_i} and the associated certified attestation Π_{U_i} with their home carrier network. These subscriber databases maintained by the cellular network provider is used to construct the Merkle tree. The first step is to identify the leaf position for the user in the tree, a VRF taking the user identifier information as the input computes the relevant position in the tree. Due to the nature of the VRF, the position for the label can be verified, and serves as an efficient way to identify the associated Merkle proof π. As shown in Figure 7.7, the data-leaves contain the information associated with the user $(U_i \| (\text{PK}_{U_i}, \Pi_{U_i})$ in addition to the commitment generated by $\text{VRF}(U_i)$, which is verifiably indexed to a leaf L_i. For U_1, this result is $\mathbb{H}(L_1)$ as shown in the figure.

Each parent node in the tree is constructed by concatenating the siblings at a level and hashing them recursively until the root. As shown in the figure, the root $\mathcal{R}$ of a cellular network with 4 subscribers is computed as $\mathbb{H}(\mathbb{H}(\mathbb{H}(L_1)\|\mathbb{H}(L_2))\|\mathbb{H}(\mathbb{H}(L_3)\|\mathbb{H}(L_4)))$ and a commitment $\sigma_{\mathcal{R}}$ on the root is generated through a cryptographic signature. The result $(\mathcal{R}, \sigma_{\mathcal{R}})$ is periodically published globally either to an append only log or a blockchain for example. Each update to the signed tree root is chained to its previous root, similar to the proposal in CONIKS [226]. To improve transparency of centrally hosted key directories, a real world implementation of VKDs has recently been implemented into WhatsApp – a widely popular centralized E2EE communication platform [216].

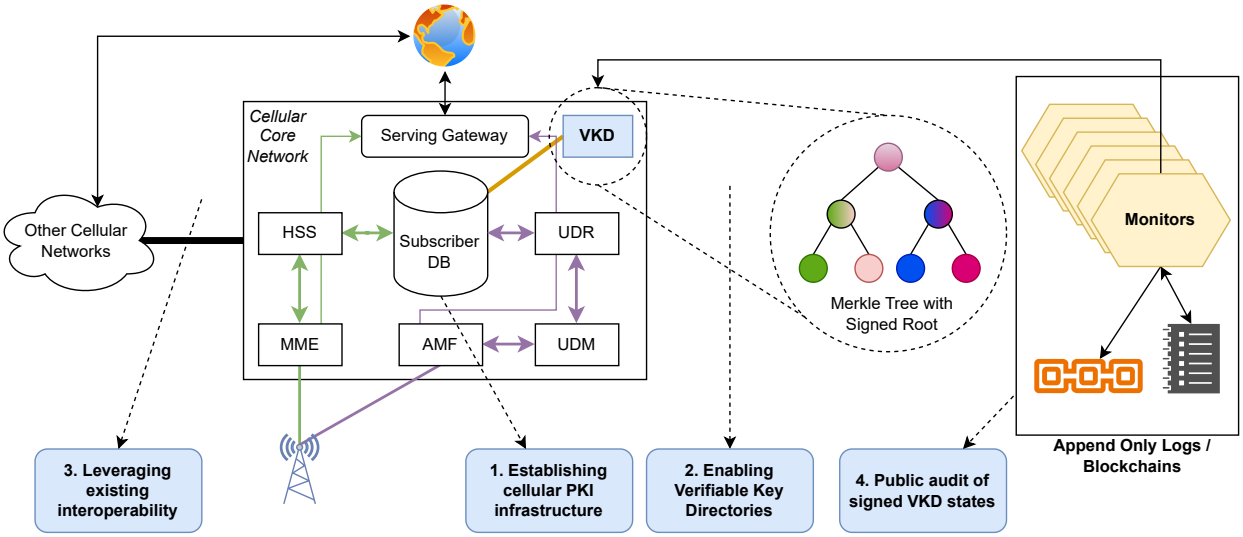


Figure 7.8: Changes required to existing 4G (green) and 5G (purple) communication flow from a carrier base station to the cellular core network. By leveraging existing interoperability between cellular networks and the Internet, establishing public key subscriber identities, and the introduction of VKDs – we enable interoperable E2EE communications with other cellular carriers and Internet based communication applications. VKDs maintain signed auditable datastructures such as Merkle trees which are globally audited and monitored by the existing certificate transparency monitors on the web or by blockchains.

As shown in Figure 7.8, the VKDs can be integrated into cellular core networks as a standalone microservice extending the core network. The introduction of the VKD services are deliberately done to be backward compatible requiring minimal changes to the cellular core maintaining the modularity of the identity and authorization mechanisms. With modifications to user devices, clients can request the cellular networks to retrieve the keying material of their intended recipient. These services can also be exposed to hidden intermediaries, who can either pass this information to the Internet entity attempting to send a secure message to the user or use the information to encrypt to only be read by the intended user preserving message confidentiality. To truly leverage the advantages of enabling verifiable key discovery and E2EE communications across providers, all participating operators must implement and deploy the VKD microservices.

Verifying the Integrity of the Information Provided

Users and clients requesting the associated information for a subscriber U_i obtain a *Merkle proof* (π) from the cellular VKD services. The Merkle proof is a sequential path from the leaf to the root of the tree consisting of intermediate nodes over which computation yields a result matching the root $\mathcal{R}$. Consider the Merkle tree shown in Figure 7.7 and a client's request for the information associated with U_2. The cellular VKD function queries the subscriber databases for the label-value pairs associated with the user obtaining the result $(\text{PK}_{U_2}, \Pi_{U_2})$ containing the public key and associated attestation. The attestation contains proof of the cellular network which initiates the creation or provision of the eSIM profile for its subscriber. The function then identifies the position of the leaf L_2 where the requested user's information is stored and retrieves the sibling at each level constructing the proof. Therefore the Merkle proof π_{U_2} consists of $\langle \mathbb{H}(L_1), \mathbb{H}(\mathbb{H}(L_3) \| \mathbb{H}(L_4)) \rangle$, and the signed root $(\mathcal{R}, \sigma_{\mathcal{R}})$. The result of the VRF indicates the index associated with the query determining the order of the concatenation.

The client given the result $(\text{PK}_{U_2}, \Pi_{U_2})$, proof π_{U_2}, the VRF commitment, and the signed root $(\mathcal{R}, \sigma_{\mathcal{R}})$ can verify the correctness of the Merkle proof and the contents by: (1) verifying the correctness of the returned public key by verifying the signed attestation with a local trust anchor such as GSMA roots, (2) verify the commitment on the root $(\mathcal{R}, \sigma_{\mathcal{R}})$ using the associated published public key of the carrier in the operator certificate, (3) verify the inclusion of the leaf at the expected index through the VRF commitment, (4) compute the leaf L_2 combining the user information with the retrieved result $(U_2 \| (\text{PK}_{U_2}, \Pi_{U_2}))$ and hash it to obtain $\mathbb{H}(L_2)$, followed by (5) traversing the proof π_{U_2} and combining each element in the traversal at the appropriate position (left/right) during concatenation with the previous hash obtained resulting in the root $\mathcal{R}'$, and finally (6) comparing the result of the traversal $\mathcal{R}'$ with $\mathcal{R}$. In the example shown in Figure 7.7, the traversal of the proof chain $\pi_{U_2} = \langle \mathbb{H}(L_1), \mathbb{H}(\mathbb{H}(L_3) \| \mathbb{H}(L_4)) \rangle$ using L_2 results in the intermediate step $\mathbb{H}(\mathbb{H}(L_1) \| \mathbb{H}(L_2))$. This result is concatenated with the next element of the proof

$\mathbb{H}(\mathbb{H}(L_3)||\mathbb{H}(L_4))$ resulting in the client computing $\mathcal{R}' = \mathbb{H}(\mathbb{H}(\mathbb{H}(L_1)||\mathbb{H}(L_2))||\mathbb{H}(\mathbb{H}(L_3)||\mathbb{H}(L_4)))$ which matches the root $\mathcal{R}$ validating the integrity and correctness of the returned response from the cellular operators involved.

Retrofitting to Existing Transparency Infrastructure

Implicit trust in cellular network operated identity infrastructure can be reduced by deploying VKD services over the subscriber databases which guarantee integrity of the data stored. However, without any external auditors, cellular networks can construct multiple Merkle trees presenting multiple valid states to the users with valid roots. As a result, operators produce verifiable commitments periodically and share the signed roots over an append only log or a blockchain network. Append only logs, such as CT logs, enable monitors to validate the correctness of the VKD root constructions. Additionally, monitors expose the functionality to act as witnesses from clients who request the monitors to perform integrity checks on the Merkle tree state constructed by the cellular networks. In a decentralized cellular network or those using blockchains for auditability, the signed tree roots of the Merkle tree states are published over the blockchain chaining the current state to the previous state. Clients additionally monitor their own key bindings to ensure that their keys have not been maliciously updated.

7.2.5 Establishing Secure E2EE Communications

Cellular networks enable two key types of communications benefiting the end users: (1) person to person (P2P) messaging, where two independent users of the cellular network communicate between themselves, and (2) application to person (A2P) messaging, where Internet enabled applications communicate with their users using a cellular identifier such as the phone number. SMS continues to be the most popular mechanism in which messages are exchanged due to its ubiquity and immediate backward compatible nature. With the establishment of VKDs, it is now

possible to establish secure E2EE communications between two users or from the applications. The recent advances in messaging protocols such as the usage of Rich Communication Services (RCS) can also leverage the advantages of transparency and auditability provided by VKDs. Below, we present the role of the VKD in enabling secure communications in both P2P and A2P contexts.

Securing Person-to-Person (P2P) Messaging

P2P messaging is a common use case in cellular networks, the SMS capabilities and relevant infrastructure in the core network have existed since the initial 2G networks and have seen minimal changes until the recent development of RCS. Below, we consider the case where two users would like to establish secure E2EE messaging between themselves solely using their cellular carrier established identities.

Consider the scenario where the users Alice, with long term identity key pairs $(\mathrm{SK}_A, \mathrm{PK}_A)$, and Bob, with identity $(\mathrm{SK}_B, \mathrm{PK}_B)$, would like to communicate securely with each other. Each user registers their associated public key identity and attestation with their respective cellular networks who include their information in a VKD. Alice initiates the communication with her cellular provider requesting Bob's associated keys who requests Bob's carrier network using the identity prefix present in the phone number to obtain the state of the VKD and associated Merkle proof. This information is optionally validated by the cellular network and forwarded to Alice who validates the integrity of the key material distributed.

The secure establishment of long term public keys with both parties enables the parties to run a *key agreement* protocol allowing the establishment of a shared secret that can be used for further communications. While this approach is a naïve mechanism to establish E2EE communication, the replacement of long term public keys with key bundles consisting of the long term public keys, medium term signed prekey pairs and some ephemeral prekey pairs enables clients to use the widely popular and secure libsignal protocol [219, 266].

Securing Application-to-Person (A2P) Messaging

The establishment of cellular VKDs opens up opportunities for Internet enabled businesses to communicate securely with intended recipients. Operators providing SMS capabilities to the Internet businesses obtain the key bundles associated with the user's carrier registered identity.

Consider the scenario where a hospital would like to send a reminder to a patient about their upcoming appointment. The hospital's appointment system integrates with a SMS gateway with whom they have explicit business partnerships. The hospital's systems craft a reminder message m for the recipient i, and send this message (i, m) over a secure TLS connection to the SMS gateway operator. The SMS gateway identifies the metadata such as the carrier information, associated with the user and obtains the user i's key bundle, VRF commitment, and a Merkle proof from the VKD services of the carrier network. This information is verified, and an encrypted message is sent to the recipient, who can successfully decrypt the message on their device thus ensuring that sensitive communications are not stored at each hop along the message transmission path.

While the scenario presented above does not require any changes to the hospital's systems responsible for sending out these reminders, it places trust in the operational correctness of the operations of the SMS gateway to not peek or store the communications in plain-text. This trust could be further reduced by indicating to the gateway about the intent to send a message to a user i, and obtaining the required proofs and information from the cellular VKDs. This information containing the VRF commitment, associated public keys or key bundles, Merkle proofs, and signed roots can be validated directly by the hospital's systems with an updated trust anchor containing a trusted certificate attesting the signing cellular network operator. As a result, the hospital systems can only reveal the encrypted cipher text associated with the original message $\mathrm{Enc}(m, \mathrm{PK}_i)$ to the SMS gateway, preventing the gateway from learning about the plaintext content of the messages being relayed – reducing the implicit trust placed in the operator and securing communications.

7.2.6 Establishing E2EE Communication Interoperability

The deployment of VKDs in cellular network cores, and in the infrastructure managed by centralized E2EE communication providers like WhatsApp and Signal, allows users with different established public key identities to securely discover other users based on a *label* associated with their key materials. These labels could be phone numbers, usernames or specialized labels within a namespace such as user-label@whatsapp.com.. The combination of an Internet based discovery services for providers, and existing cellular signaling and routing protocols allows interoperability between identity service providers. This architecture facilitates the development of end user communication applications and enables operating system vendors to provide upgraded messaging applications by default without requiring users to install applications. However, this requires all parties to adopt standardized and federated protocols allowing for algorithm agility. For example, Telegram with its MTProto algorithm would not be able to interoperate with applications based on the libsignal protocol.

While the usage of standardized mechanisms to maintain VKDs are implemented by core networks addresses the challenges with interoperability of key discovery mechanisms across providers, applications such as the standard messaging app on user devices, or 3rd party applications outside the cellular ecosystem such as WhatsApp, Telegram, or Signal must agree and support a mandatory-to-implement set of algorithms [159, 297].

7.2.7 Implications of VKDs in Cellular Network Architectures

The deployment of transparency infrastructure such as VKDs by the cellular network provider which interoperates with existing VKDs such as the ones deployed by WhatsApp allow operators to enable end-to-end encrypted communications by customers over their services by default. This allows subscribers to continue to trust their operator to route their network packets correctly but removes the need to trust them with the contents of their data. These actions improve

operator reputation and reduce the need for explicit privacy policies regarding the plaintext nature of communications, who can observe these communications, and when. On the flip side however, enabling E2EE communications and establishing transparency infrastructure preventing equivocation attacks poses challenges for existing law enforcement and national intelligence efforts that rely on cellular carriers. Further decentralization of the network providers and improving interoperability through extensive standardization further exacerbates law enforcement efforts.

The establishment of VKDs in cellular networks enables an ecosystem where Internet applications can continue to rely on the identity property they expect from cellular networks. However, it allows the applications to trust the network operator for routing information within their networks, without needing to provide the exact information. A challenge in usable security is how account recoveries and loss of keys can be managed as the burden of key recovery or changes as a result of this verifiable infrastructure lies on the cellular providers who validate users and allow them to reset their identities in an out-of-band process.

Monitors and transparency infrastructure, such as certificate transparency, on the other hand, are altruistically operated. The integration of the cellular ecosystem with the existing web ecosystem increases the load on such transparency infrastructure. Monitors doubling as witnesses, who receive requests from users to check the integrity of their information as a result are vulnerable to denial of service attacks and should be mitigated in ways similar to how CDN networks or large cloud providers address these challenges.

7.2.8 Conclusion

The establishment and enrollment of cellular networks into existing transparency infrastructure would make a significant portion of today's critical societal scale computing infrastructure auditable. While the arguments for migrating key transparency infrastructure to cellular networks is compelling, there are many technical, legal, and policy challenges that remain to be addressed.

The establishment of VKDs at the core identity layer managed by cellular networks today enable transparency and allow democratization of end user applications – satisfying the regulatory requirements set forth by law such as the European Union's Digital Markets Act [278]. The introduction of this infrastructure reduces the implicit trust placed by a user in their network operator, improves operator trustworthiness, and shifts the traditionally closed source and proprietary ecosystems into open and publicly audited ones, continuing to blur the boundaries between cellular and global Internet networks.

Chapter 8

Concluding Thoughts

A significant portion of the networking infrastructure used by billions of people today is hidden, and critical in daily life, yet it is implicitly trusted to do the right thing – enable, secure and keep user communications private. However, this implicitly trusted infrastructure globally lacks security and privacy. Access providers to the networks control how and what portion of the networks users can access while controlling their digital identity. These providers also double as connectivity providers who enable users access to the Internet and control their ability to reach their desired content, and shape the minimum security level, and privacy of their connection by deploying hidden infrastructure such as DNS resolvers and DHCP servers that automatically provide configuration information allowing clients to access the network. These providers are privy to sensitive user communications on both the Internet and in the various cellular network provided services. Lastly, Internet enabled applications today such as E2EE communication systems, ride sharing platforms, or entertainment platforms require users to provide identification details such as phone numbers – managed by cellular networks to prevent abuse such as sybil attacks to their systems, indicating the heavy reliance and trust in cellular carriers.

In this thesis, I argue that the boundaries between cellular networks and Internet service providers are no longer distinct and greatly overlap – meriting their combined study rather

than individually. Users implicitly trust cellular networks to provide connectivity services and access to the Internet and as a result inadvertently delegate the ability to manage their digital identity to their carrier networks. While communications on the Internet enabled by IP gateways managed by cellular providers are increasingly becoming secure between the client and the end host due to the usage of secure protocols such as TLS, cellular networks have not yet adopted such protocols and are vulnerable to various attacks affecting user security and privacy. So while users interact securely with an application on the Internet, any communications between the application and the user devices sent via the cellular network's core services such as SMS messages or phone calls are present in clear-text with all intermediaries along the path of the data transmissions. Despite such communications involving sensitive information such as tokens for password resets, healthcare reminders, or private communications between two users, cellular network operators have complete access to this information and are trusted to not-read the data due to the establishment of legally binding privacy policies.

Similarly, cellular network providers operating in different countries have different legal and regulatory policies they follow. These policies such as content filtering guidelines, or ability for a national intelligence agency to intercept communications either covertly or on legal grounds raise security and privacy risks for their users. Users are typically unaware of such practices as no signals are sent to the users when their communications are intercepted. Additionally, in countries with heavy Internet censorship, the access providers enforce policies preventing users from accessing secure Internet services – potentially even isolating the country's networks entirely. Users as a result observe failed message delivery, such as the OTP to sign up on an Internet enabled E2EE communication platform, or even connectivity to the global Internet.

In this thesis, I challenge the current levels of implicit trust placed in cellular network infrastructure and argue that (1) decentralization of the cellular network infrastructure among a larger set of private and smaller distributed core networks would allow greater agility and

choice for users to choose between carriers, (2) make a case that the very crux of today's Internet relying on non-transparent and non-auditable cellular identity infrastructure must be modified enabling cryptographically verifiable identities. These changes coupled with the advantages of interoperability, and routing already enabled by cellular and Internet networks have far reaching implications to how users present their identities to Internet applications, and in-turn how these applications consume these identities to enable secure communications. These changes, if done right, remove the need for cellular network operators to guarantee user privacy through privacy policy based approaches – providing technical guarantees about privacy and improve operator trustworthiness since identity and connectivity information is auditable, while removing any plaintext communications through the network.

This thesis work, also shows that in addition to the cellular networks, users implicitly trust various hidden components enabling today's usable Internet such as DNS resolvers, and certificate authorities. DNS resolvers operated within the network boundaries such as the cellular network's or the ISP's, are privy to all client traffic revealing a large amount of information about the user's browsing and navigation patterns. These communications are additionally plaintext in nature allowing active and passive adversaries along the network path to modify, and observe the communications. The usage of secure transport protocols while addressing the challenge with network adversaries continues to reveal information about the user queries to the DNS infrastructure providers. With network architectures such as Oblivious DNS over HTTPS, Internet users can distribute their trust among non-colluding entities in the network – improving their privacy.

The growing colocation of the Internet among a few large cloud providers and CDN networks while enabling easy scalability, raise privacy concerns because similar to cellular networks, users trust a small number of entities globally with massive amounts of their data. However, I argue that this colocation enables opportunities to improve user privacy, optimize resource consumption, and

push for greater transparency. In this thesis I show that, while such colocation resulted in more secure communications with end hosts since the responsibility for security is delegated to a few large providers, there exists a critical and trusted portion of the Internet which forms its long tail and does not have the same levels of security. My work identifies government infrastructure as one such example, forming the long tail of the Internet and attempts to measure, and collaboratively fix various security challenges. Despite the improvements to privacy and security to various hidden Internet infrastructure, the last part of this thesis argues that users implicitly trust the operational correctness of these services. Given the critical nature of these services, such as secure DNS resolvers, I show through measurement that resolvers tend to be misconfigured, and do not protect users from navigating to insecure resources. This work presents a highly practical and immediately deployable system focused on improving the transparency of the DNS ecosystem by providing users proof of DNS resolution, guaranteeing the operational correctness of the services, integrity of the answers, in addition to privacy.

In conclusion, this thesis questions and attempts to understand the nature of implicit trust users place on various layers of the network infrastructure. It challenges the implicit trust placed in identity and access services provided by cellular networks, and improves trust in operators through infrastructure decentralization, and cryptographic mechanisms. It also challenges the implicit trust placed in hidden network infrastructure such as DNS which is critical for a usable Internet, identifies and addresses privacy challenges by distributing trust among entities, and enables transparency to prove the operational correctness of this infrastructure. This thesis advocates for establishing a secure, private, and transparent Internet of the future where users have choice, are guaranteed secure and private access, and have verifiable guarantees of correctness provided by the infrastructure they interact with unknowingly. I acknowledge that a common (un)intended consequence of the works presented in this thesis is the increased difficulty for law enforcement on the Internet – while important, is at odds with privacy, which I firmly believe is paramount.

Acknowledgments

I am extremely grateful to my advisors Kurtis Heimerl, and Richard Anderson who took a chance on me and gave me this incredible opportunity. Their guidance, mentorship and support had a significant role in me pursuing interesting research challenges and making this work possible. I am grateful for all the independence, and freedom they have given me during this PhD journey; allowing me to work on what I truly felt was important and supporting me through it. Over the last few years, I've had the incredible opportunity to learn from you, and get to know you – both professionally and personally and I'll be carrying these lessons with me for a really long time.

I would like to express my deepest appreciation to my thesis committee members Arvind Krishnamurthy, Ryan Calo and Tadayoshi Kohno for their valuable feedback. During my time here, I've had the opportunity to learn from Arvind and Yoshi's lectures during class which provided the required background and influenced my thesis direction. I am grateful for the opportunities of engaging with Ryan and the Tech Policy Lab kindling my interest in the legal and policy domain and allowing me to recognize the implications of my work.

Words cannot express my gratitude to the brilliant teams and collaborators at Cloudflare who have been extremely influential and played a key role in shaping my thesis direction. As I look back at my time at Cloudflare and the long term collaboration over the years, I recognize that I've had many champions – Marwan Fayed who has been an excellent mentor, encouraged the

wildest of ideas, and challenged me at every step constantly pushing me towards rigorous and high standards while being extremely kind and patient; a quality I deeply admire and aspire to embody in my own actions. I'd also like to thank Chris Wood, and Nick Sullivan for taking a chance on me, teaching me, and encouraging me at every step of the way. I am incredibly grateful for the conversations and learning opportunities that everyone else on the research, DNS, protocols, and speed teams gave me – Andrew Galloni, Avani Wildani, Bas Westerbaan, Bob Halley, Cefan Rubin, Chris Patton, Christian Elmerot, Jonathan Hoyland, Kristin Berdan, Lucas Pardue, Luke Valenta, Marek Vavruša, Peter Wu, Suleman Ahmad, Suphanat Chunhapanya, Tanya Verma, Tara Whalen, Thibault Meunier, Vânia Gonçalves, Vasilis Giotsas, Wesley Evans, and Yevgen Safronov.

I'd like to thank my collaborators at CyberGreen whose support empowered me to define and take charge of the direction and deployment of various Internet scanning infrastructure. I thank Adam Lange, Adam Shostack, Arastoo Taslim, Jonathan de Koning, Jarrod O'Malley, and Yurie Ito for trusting me and giving me the freedom and flexibility required to execute various research ideas. I'd also like to thank my collaborators at Microsoft Research's Systems Security and Cryptography groups – Esha Ghosh, Sebastian Angel, Srinath Setty, and Weidong Cui for encouraging me and piquing my interest in confidential, verifiable, and transparent infrastructure.

Pursuing a doctoral degree was never something I intended to do. The time I spent as a Research Fellow at Microsoft Research India working with Bill Thies, Muthian Sivathanu, and Satya Lokam changed that. I am immensely grateful to Bill for introducing me to research, taking the chance on me and teaching me about it, showing me what a career in research would look like, and inspiring me during our collaboration, and championing me ever since. I am deeply indebted to Muthian Sivathanu for all the advice, challenging me to improve and shaping the directions as I headed towards a PhD program. I would like to express my deepest gratitude to Satya Lokam whose collaboration, constant encouragement, motivation, and advice over the many years has had a significant impact in me choosing and exploring interdisciplinary research directions.

During my time here, I've had amazing lab mates, colleagues, and friends who stuck with me through thick and thin, celebrated my successes and failures, encouraged me, and played a key role in helping me grow – both personally and professionally. I'd like to thank my lab mates Ananditha Raghunath, Emmanuel Azuh Mensah, Esther Jang, Innocent Obi, Lisa Orii, Matthew Johnson, Matthew Ziegler, Nick Durand, Nussara Tieanklin, Samia Ibtasam, Spencer Sevilla, and Waylon Brunette. I'd like to thank my collaborators, and the community of researchers working in the areas of systems, security, accessibility, and human computer interaction, and AI/ML research, who have provided valuable and critical feedback to my work – Aditya Kusupati, Alex Mwotil, Amar Budhiraja, Anant Mittal, Anna Simpson, Ashrujit Ghoshal, Asit Kadayan, Bandhav Veluri, Ben Weintraub, Deepak Maram, Dhruv Jain, Eric Zeng, Franzi Roesner, Galen Weld, Gautam Akiwate, Jason Hoffman, Jennifer Mankoff, Jialin Li, Kaiming Cheng, Kate Glazko, Katie Lim, Kentrell Owens, Marissa Radensky, Miranda Wei, Motoya Ohnishi, Nirmalendu Diwakar, Petros Gigis, Prashanth Rangarajan, Pratyush Patel, Rachel Hong, Raghav Somani, Ram Sundara Raman, Ratul Mahajan, Sambhav Satija, Stefano Tessaro, Sunil Bajpai, Talha Paracha, Tina Yeung, Tusher Chakraborty, Umar Iqbal, Vishwas Sathish, Yael Eiger, and Zihao Ye.

I'd like to thank the incredible support from the graduate advising team – Chris Nagle, Elise DeGoede, Joe Eckert, and Les Sessoms; who have navigated the strangest of challenges that I faced and bore the brunt of various bureaucratic processes. During the time here, I've had the opportunity to work with brilliant students who supported my research and contributed to it significantly – Abhishek Shah, Akhila Narayanan, Darren Denq, Donna Albee, Evan Lam, Mark Theeranantachai, Mohan Kukreja, Rachel Alwan, Todd Meng, and Zhennan Zhou.

I am incredibly grateful for my closest friends – Gopal Singh Meena, Priyal Suneja, Rahul Jain, Sreerag Sreenath, and Venkatesh Potluri; who have been my unwavering and unhesitating support system that I could fall back on through this journey – both professionally and personally. Thank you for patiently listening to me, and motivating me through the hardest of times.

Finally and above all, I extend my heartfelt gratitude for my family – Jyothi Sreeramadasu (*mother*), and Subramanyam Singanamalla (*father*) whose endless patience with my long hours of work, sacrifices, and belief in my abilities gave me the strength to persevere in the face of challenges. This achievement is as much yours as it is mine.

Thank you for believing in me!

Bibliography

[1] European Telecommunications Standards Institute (ETSI). *ETSI TS 135 206 - Specification of Milenage Algorithm Set*. https://www.etsi.org/deliver/etsi_ts/135200_135299/135206/09.00.00_60/ts_135206v090000p.pdf. 2010.

[2] Patrick Nohe (GlobalSign). *One Year Certificates - Maximum SSL/TLS Certificate Validity is Now One Year*. https://casecurity.org/2020/07/09/one-year-certs/. (Accessed on 09/07/2020). July 2020.

[3] Seattle Community Network (SCN). *Seattle Community Network*. https://seattlecommunitynetwork.org/. (Accessed on 02/21/2023). Feb. 2023.

[4] 3GPP. *3GPP Mobile Broadband Standard - Release 14*. https://www.3gpp.org/release-14. 2017.

[5] Donald E. Eastlake 3rd and Charles W. Kaufman. *Domain Name System Security Extensions*. RFC 2065. Jan. 1997.
DOI: 10.17487/RFC2065.

[6] *95% of HTTPS servers vulnerable to trivial MITM attacks | Netcraft News*. https://news.netcraft.com/archives/2016/03/17/95-of-https-servers-vulnerable-to-trivial-mitm-attacks.html. (Accessed on 09/11/2020). Mar. 2016.

[7] Josh Aas. *Preparing to Issue 200 Million Certificates in 24 Hours*. https://letsencrypt.org/2021/02/10/200m-certs-24hrs.html. Feb. 2021.

[8] Josh Aas, Richard Barnes, Benton Case, Zakir Durumeric, Peter Eckersley, Alan Flores-López, J Alex Halderman, Jacob Hoffman-Andrews, James Kasten, Eric Rescorla, et al. "Let's Encrypt: an automated certificate authority to encrypt the entire web". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 2019, pp. 2473–2487.

[9] Joe Abley, Jakob Schlyter, Guillaume Bailey, and Paul E. Hoffman. *DNSSEC Trust Anchor Publication for the Root Zone*. RFC 7958. Aug. 2016.
DOI: 10.17487/RFC7958.

[10] Mustafa Emre Acer, Emily Stark, Adrienne Porter Felt, Sascha Fahl, Radhika Bhargava, Bhanu Dev, Matt Braithwaite, Ryan Sleevi, and Parisa Tabriz. "Where the Wild Warnings Are: Root Causes of Chrome HTTPS Certificate Errors". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. CCS '17. Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 1407–1420.
DOI: 10.1145/3133956.3134007.

[11] Michael Adeyeye and Paul Gardner-Stephen. "The Village Telco project: a reliable and practical wireless mesh telephony infrastructure". In: *EURASIP Journal on Wireless Communications and Networking* 2011.1 (2011), p. 78.

[12] Neil Agarwal, Matteo Varvello, Andrius Aucinas, Fabián Bustamante, and Ravi Netravali. "Mind the Delay: The Adverse Effects of Delay-Based TCP on HTTP". In: *Proceedings of the 16th International Conference on Emerging Networking EXperiments and Technologies*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 364–370. ISBN: 9781450379489.
ISBN: 9781450379489.

[13] Devdatta Akhawe, Bernhard Amann, Matthias Vallentin, and Robin Sommer. "Here's My Cert, so Trust Me, Maybe? Understanding TLS Errors on the Web". In: *Proceedings of the 22nd International Conference on World Wide Web*. WWW '13. Rio de Janeiro, Brazil: Association for Computing Machinery, 2013, pp. 59–70.
DOI: 10.1145/2488388.2488395.

[14] Nadem Alfardan, Daniel J. Bernstein, Kenneth G. Paterson, Bertram Poettering, and Jacob C.N. Schuldt. "On the Security of RC4 in TLS and WPA". In: *USENIX Security*. 2013.

[15] Kamal Alieyan, Mohammed M Kadhum, Mohammed Anbar, Shafiq Ul Rehman, and Naser KA Alajmi. "An overview of DDoS attacks based on DNS". In: *2016 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE. 2016, pp. 276–280.

[16] NGMN Alliance. "Description of network slicing concept". In: *NGMN 5G P* 1.1 (2016), pp. 1–11.

[17] Stephanie Alt, Pierre-Alain Fouque, Gilles Macario-Rat, Cristina Onete, and Benjamin Richard. "A cryptographic analysis of umts/lte aka". In: *Applied Cryptography and Network Security: 14th International Conference, ACNS 2016, Guildford, UK, June 19-22, 2016. Proceedings*. Springer. 2016, pp. 18–35.

[18] Johanna Amann, Oliver Gasser, Quirin Scheitle, Lexi Brent, Georg Carle, and Ralph Holz. "Mission accomplished? HTTPS security after DigiNotar". In: *Proceedings of the 2017 Internet Measurement Conference*. 2017, pp. 325–340.

[19] Marios Anagnostopoulos, Georgios Kambourakis, Elisavet Konstantinou, and Stefanos Gritzalis. "DNSSEC vs. DNSCurve: A side-by-side comparison". In: *Situational Awareness in Computer Network Defense: Principles, Methods and Applications*. IGI Global, 2012, pp. 201–220.

[20] Marios Anagnostopoulos, Georgios Kambourakis, Panagiotis Kopanos, Georgios Louloudakis, and Stefanos Gritzalis. “DNS amplification attack revisited”. In: *Computers & Security* 39 (2013), pp. 475–485.

[21] Amelia Andersdotter and Signatories. *A letter about Google AMP*. http://ampletter.org/. Jan. 2018.

[22] Cristiano Antonelli. “Technological change and multinational growth in international telecommunications services”. In: *Review of Industrial Organization* 10 (1995), pp. 161–180.

[23] Jacob Appelbaum. “Detecting certificate authority compromises and web browser collusion”. In: *Tor Blog* 22 (2011).

[24] Apple. *List of available trusted root certificates in MacOS*. https://support.apple.com/en-gb/HT208127. (Accessed on 06/02/2020). June 2018.

[25] Roy Arends, R Austein, M Larson, Daniel Massey, and Scott W Rose. “Protocol modifications for the DNS security extensions RFC 4035”. In: (2005).

[26] Roy Arends, Geoffrey Sisson, David Blacka, and Ben Laurie. *DNS Security (DNSSEC) Hashed Authenticated Denial of Existence*. RFC 5155. Mar. 2008.
DOI: 10.17487/RFC5155.

[27] Suranjith Ariyapperuma and Chris J. Mitchell. “Security vulnerabilities in DNS and DNSSEC”. In: *The Second International Conference on Availability, Reliability and Security (ARES’07)*. Vienna, Austria: IEEE, 2007, pp. 335–342.
DOI: 10.1109/ARES.2007.139.

[28] Cory L Armstrong. “Providing a clearer view: An examination of transparency on local government websites”. In: *Government Information Quarterly* 28.1 (2011), pp. 11–16.

[29] Canadian Internet Registration Authority. *CIRA Canadian Shield | Free public DNS for Canadians*. en. 2024.
URL: https://www.cira.ca/en/canadian-shield/.

[30] Catalin-Alexandru Avram, Kenneth Salem, and Bernard Wong. “Latency amplification: Characterizing the impact of web page content on load times”. In: *2014 IEEE 33rd International Symposium on Reliable Distributed Systems Workshops*. IEEE. 2014, pp. 20–25.

[31] Michael Backes, Aniket Kate, Praveen Manoharan, Sebastian Meiser, and Esfandiar Mohammadi. “AnoA: A Framework for Analyzing Anonymous Communication Protocols”. In: *2013 IEEE 26th Computer Security Foundations Symposium*. 2013, pp. 163–178.
DOI: 10.1109/CSF.2013.18.

[32] Joonsang Baek and Yuliang Zheng. “Identity-based threshold decryption”. In: *International Workshop on Public Key Cryptography*. Springer. 2004, pp. 262–276.

[33] Kenji Baheux. *Chromium Blog: A safer and more private browsing experience with Secure DNS*. https://blog.chromium.org/2020/05/a-safer-and-more-private-browsing-DoH.html. (Accessed on 09/15/2020). May 2020.

[34] Wei Bai, Michael Pearson, Patrick Gage Kelley, and Michelle L Mazurek. "Improving non-experts' understanding of end-to-end encryption: an exploratory study". In: *2020 IEEE european symposium on security and privacy workshops (EuroS&PW)*. IEEE. 2020, pp. 210–219.

[35] Roger Baig, Ramon Roca, Felix Freitag, and Leandro Navarro. "Guifi. net, a crowdsourced network infrastructure held in common". In: *Computer Networks* 90 (2015), pp. 150–165.

[36] Rojeena Bajracharya, Rakesh Shrestha, and Haejoon Jung. "Future is unlicensed: Private 5G unlicensed network for connecting industries of future". In: *Sensors* 20.10 (2020), p. 2774.

[37] Fred Baker and Pekka Savola. *Ingress Filtering for Multihomed Networks*. RFC 3704. Mar. 2004.
DOI: 10.17487/RFC3704.

[38] Somprakash Bandyopadhyay, Kazuo Hasuike, S Horisawa, and S Tawara. "An Adaptive MAC Protocol for Wireless Ad Hoc Community Network (WACNet) using Electronically Steerable Passive Array Radiator Antenna". In: *Global Telecommunications Conference, 2001. GLOBECOM'01. IEEE*. Vol. 5. IEEE. 2001, pp. 2896–2900.

[39] Mary Claire Barela, Mae Sincere Blanco, Philip Martinez, Miguel Carlo Purisima, Kurtis Heimer, Matthew Podolsky, Eric Brewer, and Cedric Angelo Festin. "Towards Building a Community Cellular Network in the Philippines: Initial Site Survey Observations". In: *Proceedings of the Eighth International Conference on Information and Communication Technologies and Development*. ICTD '16. Ann Arbor, MI, USA: Association for Computing Machinery, 2016.
DOI: 10.1145/2909609.2909639.

[40] Mary Claire Barela, Josephine Dionisio, Kurtis Heimerl, Manuel Victor Sapitula, and Cedric Angelo Festin. "Connecting Communities through Mobile Networks: The VBTS-CoCoMoNets Project". In: *Global Information Society Watch 2018: Community Networks*. Ed. by Alan Finlay. Association for Progressive Communications, 2018.

[41] Paul Barford and Mark Crovella. "A performance evaluation of hyper text transfer protocols". In: *Proceedings of the 1999 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*. 1999, pp. 188–197.

[42] Elad Barkan, Eli Biham, and Nathan Keller. "Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication". In: *Advances in Cryptology - CRYPTO 2003*. Ed. by Dan Boneh. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 600–616. ISBN: 978-3-540-45146-4.
ISBN: 978-3-540-45146-4.

[43] Elaine Barker and Allen Roginsky. *SP 800-131 (June 11 2010): Recommendation for the Transitioning of Cryptographic Algorithms and Key Lengths*. https://csrc.nist.gov/csrc/media/publications/sp/800-131/archive/2010-06-16/documents/sp800-131-draft2-june2010.pdf. (Accessed on 03/06/2020). June 2015.

[44] Richard Barnes. *Use Cases and Requirements for DNS-Based Authentication of Named Entities (DANE)*. RFC 6394. Oct. 2011.
DOI: 10.17487/RFC6394.

[45] Richard Barnes, Karthikeyan Bhargavan, Benjamin Lipp, and Christopher A. Wood. *Hybrid Public Key Encryption*. Internet-Draft draft-irtf-cfrg-hpke-08. Work in Progress. Internet Engineering Task Force, Feb. 2021. 100 pp.
URL: https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-hpke-08.

[46] Richard Barnes, Karthikeyan Bhargavan, Benjamin Lipp, and Christopher A. Wood. *Hybrid Public Key Encryption*. RFC 9180. Feb. 2022.
DOI: 10.17487/RFC9180.

[47] Richard Barnes, Jacob Hoffman-Andrews, Daniel McCarney, and James Kasten. *Automatic Certificate Management Environment (ACME)*. RFC 8555. Mar. 2019.
DOI: 10.17487/RFC8555.

[48] David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, and Vincent Stettler. "A formal analysis of 5G authentication". In: *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*. 2018, pp. 1383–1396.

[49] Joseph Bebel and Dev Ojha. "Ferveo: Threshold decryption for mempool privacy in bft networks". In: *Cryptology ePrint Archive* (2022).

[50] Mike Belshe, Roberto Peon, and Martin Thomson. *Hypertext Transfer Protocol Version 2 (HTTP/2)*. RFC 7540. May 2015.
DOI: 10.17487/RFC7540.

[51] Jake A Berkowsky and Thaier Hayajneh. "Security issues with certificate authorities". In: *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*. IEEE. 2017, pp. 449–455.

[52] Tim Berners-Lee, Roy Fielding, and Henrik Frystyk. *Hypertext transfer protocol–HTTP/1.0*. 1996.

[53] Daniel J Bernstein. "DNSCurve: Usable security for DNS". In: *dnscurve.org* 4 (2009).

[54] Oliver Berthold, Hannes Federrath, and Stefan Köpsell. "Web MIXes: A system for anonymous and unobservable Internet access". In: *Designing privacy enhancing technologies*. Springer. 2001, pp. 115–129.

[55] Henry Birge-Lee, Yixin Sun, Anne Edmundson, Jennifer Rexford, and Prateek Mittal. "Bamboozling certificate authorities with {BGP}". In: *27th USENIX Security Symposium (USENIX Security 18)*. 2018, pp. 833–849.

[56] Henry Birge-Lee, Liang Wang, Daniel McCarney, Roland Shoemaker, Jennifer Rexford, and Prateek Mittal. "Experiences Deploying {Multi-Vantage-Point} Domain Validation at Let's Encrypt". In: *30th usenix security symposium (usenix security 21)*. 2021, pp. 4311–4327.

[57] Mike Bishop, Nick Sullivan, and Martin Thomson. *Secondary Certificate Authentication in HTTP/2*. Internet-Draft draft-ietf-httpbis-http2-secondary-certs-06. Work in Progress. Internet Engineering Task Force, May 2020.
URL: https://datatracker.ietf.org/doc/html/draft-ietf-httpbis-http2-secondary-certs-06.

[58] Zack Bloom. *An Update on CDNJS*. Cloudflare Blog. 2019.
URL: https://blog.cloudflare.com/an-update-on-cdnjs/.

[59] Kevin Borgolte, Tithi Chattopadhyay, Nick Feamster, Mihir Kshirsagar, Jordan Holland, Austin Hounsel, and Paul Schmitt. "How DNS over HTTPS is Reshaping Privacy, Performance, and Policy in the Internet Ecosystem". In: *Performance, and Policy in the Internet Ecosystem (July 27, 2019)* (2019).

[60] Timm Böttger, Felix Cuadrado, Gianni Antichi, Eder Leão Fernandes, Gareth Tyson, Ignacio Castro, and Steve Uhlig. "An Empirical Study of the Cost of DNS-over-HTTPS". In: *Proceedings of the Internet Measurement Conference*. 2019, pp. 15–21.

[61] Robert Braden. *RFC1122: Requirements for Internet hosts-communication layers*. 1989.

[62] Lorenz Breidenbach, Christian Cachin, Benedict Chan, Alex Coventry, Steve Ellis, Ari Juels, Farinaz Koushanfar, Andrew Miller, Brendan Magauran, Daniel Moroz, et al. "Chainlink 2.0: Next steps in the evolution of decentralized oracle networks". In: *Chainlink Labs* 1 (2021), pp. 1–136.

[63] Nevil Brownlee, Kimberly C Claffy, and Evi Nemeth. "DNS measurements at a root server". In: *GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No. 01CH37270)*. Vol. 3. IEEE. 2001, pp. 1672–1676.

[64] Eric Brunner-Williams, Bill Manning, and Donald E. Eastlake 3rd. *Domain Name System (DNS) IANA Considerations*. RFC 2929. Sept. 2000.
DOI: 10.17487/RFC2929.

[65] Michael Buettner and Kenji Baheux. *Continuing our journey to bring instant experiences to the whole web*. https://blog.chromium.org/2020/12/continuing-our-journey-to-bring-instant.html. Dec. 2020.

[66] Randy Bush and Rob Austein. *The Resource Public Key Infrastructure (RPKI) to Router Protocol*. RFC 6810. Jan. 2013.
DOI: 10.17487/RFC6810.

[67] Greta Byrum and Joshua Breitbart. "Wireless organizing in Detroit: Churches as networked sites in under–resourced urban areas". In: *First Monday* (2013).

[68] bzsat. *I am from iran, verification code doesn't send neither by sms nor call : r/signal*. https://www.reddit.com/r/signal/comments/xlznsd/i_am_from_iran_verification_code_doesnt_send/. Sept. 2022.

[69] CA Browser Forum. *Ballot 193 - 825-day Certificate Lifetimes - CAB Forum*. https://cabforum.org/2017/03/17/ballot-193-825-day-certificate-lifetimes/. (Accessed on 05/24/2020). Mar. 2017.

[70] CA Browser Forum. *Ballot SC22 - Reduce Certificate Lifetimes (v2) - CAB Forum*. https://cabforum.org/2019/09/10/ballot-sc22-reduce-certificate-lifetimes-v2/. (Accessed on 05/24/2020). Sept. 2019.

[71] Ian Carlos Campbell. *A text message routing company suffered a five-year-long breach - The Verge*. https://www.theverge.com/2021/10/6/22713543/syniverse-hack-five-years-text-messages. (Accessed on 01/18/2022). Oct. 2021.

[72] Manuel Castells. "The impact of the internet on society: a global perspective". In: *Change* 19 (2014), pp. 127–148.

[73] Sergio Castillo-Perez and Joaquin Garcia-Alfaro. "Evaluation of two privacy-preserving protocols for the DNS". In: *2009 Sixth International Conference on Information Technology: New Generations*. IEEE. 2009, pp. 411–416.

[74] Miguel Castro, Barbara Liskov, et al. "Practical Byzantine fault tolerance". In: *OSDI*. Vol. 99. 1999. 1999, pp. 173–186.

[75] Melissa Chase, Apoorvaa Deshpande, Esha Ghosh, and Harjasleen Malvai. "Seemless: Secure end-to-end encrypted messaging with less trust". In: *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*. 2019, pp. 1639–1656.

[76] A Chau and S Hertzberg. *California Consumer Privacy Act of 2018 1798.140 (v)*. https://leginfo.legislature.ca.gov/faces/codes_displaySection.xhtml?lawCode=CIV&sectionNum=1798.140.. (Accessed on 02/27/2021). 2018.

[77] Rebecca LJ Chen, Victor SC Chen, Elisa CY Su, and Yi-Hong Wang. "Introduction to IMS". In: *Business Models and Drivers for Next-Generation IMS Services* (2007), p. 59.

[78] Stuart Cheshire, Bernard D. Aboba, and Erik Guttman. *RFC 3927: Dynamic configuration of IPv4 link-local addresses*. 2005.

[79] Anne SY Cheung. "Location privacy: The challenges of mobile service devices". In: *Computer Law & Security Review* 30.1 (2014), pp. 41–54.

[80] Rishabh Chhabra, Paul Murley, Deepak Kumar, Michael Bailey, and Gang Wang. "Measuring DNS-over-HTTPS Performance around the World". In: *Proceedings of the 21st ACM Internet Measurement Conference*. IMC '21. Virtual Event: Association for Computing Machinery, 2021, pp. 351–365.
DOI: 10.1145/3487552.3487849.

[81] Cisco. *Umbrella Popularity List*. (Accessed on 09/07/2020). 2016.

[82] David Clark. "The design philosophy of the DARPA Internet protocols". In: *Symposium proceedings on Communications architectures and protocols*. 1988, pp. 106–114.

[83] Google Cloud. *App Engine Application Platform - Google Cloud*. https://cloud.google.com/appengine. (Accessed on 02/27/2021).

[84] Google Cloud. *Google Compute Engine - Machine Types*. https://cloud.google.com/compute/docs/machine-types. (Accessed on 09/16/2020).

[85] Cloudflare. *CFSSL: Cloudflare's PKI and TLS toolkit*. https://github.com/cloudflare/cfssl. 2019.

[86] Cloudflare. *Cloudflare Workers*. https://workers.cloudflare.com/. (Accessed on 09/15/2020).

[87] Cloudflare. *DNS over Tor | Cloudflare Developer Docs*. https://developers.cloudflare.com/1.1.1.1/fun-stuff/dns-over-tor/. (Accessed on 09/15/2020).

[88] Cloudflare. *Go Networking*. 2022.
URL: https://github.com/cloudflare/net-originframe.

[89] Cloudflare. *Merkle Town - Explore the Certificate Transparency Ecosystem*. https://ct.cloudflare.com/. 2021.

[90] Cloudflare. *ORIGIN Frame Implementation in Go*. 2022.
URL: https://github.com/cloudflare/go-originframe.

[91] Cloudflare. *The Cloudflare global network*. 2022.
URL: https://www.cloudflare.com/en-gb/network/.

[92] Cloudflare. *What Happens in a TLS Handshake? | SSL Handshake | Cloudflare*. https://www.cloudflare.com/learning/ssl/what-happens-in-a-tls-handshake/. (Accessed on 05/24/2020).

[93] Internet Systems Consortium. *DNSSEC and BIND 9 - ISC*. https://www.isc.org/dnssec/. (Accessed on 11/26/2023). 2023.

[94] OWASP Contributors. *Man-in-the-middle Software Attack | OWASP Foundation*. https://owasp.org/www-community/attacks/Man-in-the-middle_attack. (Accessed on 09/11/2020). July 2020.

[95] Wikipedia Article Contributors. *Phase Out | 3G*. https://en.wikipedia.org/wiki/3G#Phase-out. May 2024.

[96] Alex Cowperthwaite and Anil Somayaji. "The futility of DNSSEC". In: *Annual Symposium Information Assurance (ASIA)*. Citeseer. 2010.

[97] Cas Cremers and Martin Dehnel-Wild. "Component-Based Formal Analysis of 5G-AKA: Channel Assumptions and Session Confusion". In: *Network and Distributed Systems Security (NDSS) Symposium 2019*. Feb. 2019.
URL: https://publications.cispa.saarland/2758/.

[98] Cas Cremers, Marko Horvat, Jonathan Hoyland, Sam Scott, and Thyla van der Merwe. "A comprehensive symbolic analysis of TLS 1.3". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 2017, pp. 1773–1788.

[99] Debajyoti Das, Sebastian Meiser, Esfandiar Mohammadi, and Aniket Kate. "Anonymity Trilemma: Strong Anonymity, Low Bandwidth Overhead, Low Latency - Choose Two". In: *2018 IEEE Symposium on Security and Privacy (SP)*. 2018, pp. 108–126.
DOI: 10.1109/SP.2018.00011.

[100] Debajyoti Das, Sebastian Meiser, Esfandiar Mohammadi, and Aniket Kate. "Comprehensive Anonymity Trilemma: User Coordination is not enough". In: *Proceedings on Privacy Enhancing Technologies* 2020.3 (2020), pp. 356–383.
DOI: doi:10.2478/popets-2020-0056.

[101] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. "Privacy pass: Bypassing internet challenges anonymously". In: *Proceedings on Privacy Enhancing Technologies* 2018.3 (2018), pp. 164–180.

[102] Ryan Daws. "Vodafone UK completes 3G shutdown to enhance 4G/5G services". In: *Telecoms Tech News* (Feb. 2024).
URL: https://www.telecomstechnews.com/news/2024/feb/28/vodafone-uk-3g-shutdown-enhance-4g-5g-services/.

[103] Selena Deckelmann. *Firefox continues push to bring DNS over HTTPS by default for US users - The Mozilla Blog*. https://blog.mozilla.org/blog/2020/02/25/firefox-continues-push-to-bring-dns-over-https-by-default-for-us-users/. (Accessed on 09/15/2020). Feb. 2020.

[104] Frank Denis. *Anonymized DNSCrypt Specification*. https://github.com/DNSCrypt/dnscrypt-protocol/blob/master/ANONYMIZED-DNSCRYPT.txt. (Accessed on 09/15/2020). June 2020.

[105] Frank Denis and Contributors. *A flexible DNS proxy, with support for encrypted DNS protocols*. https://github.com/DNSCrypt/dnscrypt-proxy/. (Accessed on 09/17/2020).

[106] Apple Developer. *DNS Proxy Provider | Apple Developer Documentation*. https://developer.apple.com/documentation/networkextension/dns_proxy_provider. (Accessed on 09/15/2020).

[107] Apple Developer. *Enable encrypted DNS - WWDC 2020*. https://developer.apple.com/videos/play/wwdc2020/10047/. (Accessed on 09/15/2020).

[108] Google DNS. *Your Privacy - Public DNS - Google Developers*. https://developers.google.com/speed/public-dns/privacy. (Accessed on 02/27/2021).

[109] DNSLink Community and Protocol Labs. *DNSLink Standard*. [Online; accessed 11. Feb. 2023]. Sept. 2022.
URL: https://www.dnslink.io/.

[110] DNSViz. *brokendnssec.net*. https://dnsviz.net/d/brokendnssec.net/dnssec/. (Accessed on 05/27/2023). Apr. 2023.

[111] Danny Dolev and Andrew Yao. "On the security of public key protocols". In: *IEEE Transactions on information theory* 29.2 (1983), pp. 198–208.

[112] DotGov. *Making .gov More Secure by Default | DotGov*. https://home.dotgov.gov/management/preloading/dotgovhttps/. (Accessed on 09/12/2020). June 2020.

[113] Ralph Droms. *RFC2131: Dynamic Host Configuration Protocol*. 1997.

[114] Viktor Dukhovni and Wes Hardaker. *The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance*. RFC 7671. Oct. 2015. DOI: 10.17487/RFC7671.

[115] Viktor Dukhovni, Shumon Huque, Willem Toorop, Paul Wouters, and Melinda Shore. *TLS DNSSEC Chain Extension*. RFC 9102. Aug. 2021. DOI: 10.17487/RFC9102.

[116] Zakir Durumeric, David Adrian, Ariana Mirian, Michael Bailey, and J. Alex Halderman. "A Search Engine Backed by Internet-Wide Scanning". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. CCS '15. Denver, Colorado, USA: Association for Computing Machinery, 2015, pp. 542–553. DOI: 10.1145/2810103.2813703.

[117] Zakir Durumeric, David Adrian, Ariana Mirian, James Kasten, Elie Bursztein, Nicolas Lidzborski, Kurt Thomas, Vijay Eranti, Michael Bailey, and J. Alex Halderman. "Neither Snow Nor Rain Nor MITM...: An Empirical Analysis of Email Delivery Security". In: *Proceedings of the 2015 Internet Measurement Conference*. IMC '15. Tokyo, Japan: Association for Computing Machinery, 2015, pp. 27–39. DOI: 10.1145/2815675.2815695.

[118] Zakir Durumeric, James Kasten, Michael Bailey, and J Alex Halderman. "Analysis of the HTTPS certificate ecosystem". In: *Proceedings of the 2013 conference on Internet measurement conference*. 2013, pp. 291–304.

[119] Zakir Durumeric, Zane Ma, Drew Springall, Richard Barnes, Nick Sullivan, Elie Bursztein, Michael Bailey, J Alex Halderman, and Vern Paxson. "The Security Impact of HTTPS Interception." In: *NDSS*. 2017.

[120] Zakir Durumeric, Eric Wustrow, and J Alex Halderman. "ZMap: Fast Internet-wide Scanning and Its Security Applications." In: *USENIX Security Symposium*. Vol. 8. 2013, pp. 47–53.

[121] Kit Eaton. *How One Second Could Cost Amazon USD 1.6 Billion In Sales*. https://www.fastcompany.com/1825005/how-one-second-could-cost-amazon-16-billion-sales. Mar. 2012.

[122] Lilian Edwards. "Content filtering and the new censorship". In: *2010 Fourth International Conference on Digital Society*. IEEE. 2010, pp. 317–322.

[123] Electronic Frontier Foundation EFF. *Certbot*. https://certbot.eff.org/. (Accessed on 09/12/2020).

[124] ENS Community. *Ethereum Name Service*. en. ENS Labs Ltd., 2022. URL: https://docs.ens.domains/.

[125] Tariq Fadai, Sebastian Schrittwieser, Peter Kieseberg, and Martin Mulazzani. “Trust me, I’m a Root CA! Analyzing SSL Root CAs in Modern Browsers and Operating Systems”. In: *2015 10th International Conference on Availability, Reliability and Security*. IEEE. 2015, pp. 174–179.

[126] Marwan Fayed, Lorenz Bauer, Vasileios Giotsas, Sami Kerola, Marek Majkowski, Pavel Odintsov, Jakub Sitnicki, Taejoong Chung, Dave Levin, Alan Mislove, Christopher A. Wood, and Nick Sullivan. “The Ties That Un-Bind: Decoupling IP from Web Services and Sockets for Robust Addressing Agility at CDN-Scale”. In: *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*. SIGCOMM ’21. Virtual Event, USA: ACM, 2021, pp. 433–446. DOI: 10.1145/3452296.3472922.

[127] Hannes Federrath, Karl-Peter Fuchs, Dominik Herrmann, and Christopher Piosecny. “Privacy-preserving DNS: analysis of broadcast, range queries and mix-based protection methods”. In: *European Symposium on Research in Computer Security*. Springer. 2011, pp. 665–683.

[128] Adrienne Porter Felt, Richard Barnes, April King, Chris Palmer, Chris Bentzel, and Parisa Tabriz. “Measuring HTTPS Adoption on the Web”. In: *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, Aug. 2017, pp. 1323–1338. ISBN: 978-1-931971-40-9.
ISBN: 978-1-931971-40-9.

[129] Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, and Tim Berners-Lee. *Hypertext transfer protocol–HTTP/1.1*. 1999.

[130] Michèle Finck and Frank Pallas. “They who must not be identified-distinguishing personal from non-personal data under the GDPR”. In: *International Data Privacy Law* 10.1 (Mar. 2020), pp. 11–36.
DOI: 10.1093/idpl/ipz026.

[131] Tobias Flach, Nandita Dukkipati, Andreas Terzis, Barath Raghavan, Neal Cardwell, Yuchung Cheng, Ankur Jain, Shuai Hao, Ethan Katz-Bassett, and Ramesh Govindan. “Reducing web latency: the virtue of gentle aggression”. In: *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM*. 2013, pp. 159–170.

[132] Frank Denis and Yecheng Fu. *DNSCrypt: A protocol to improve DNS security*. https://www.dnscrypt.org/. (Accessed on 02/20/2021). Feb. 2021.

[133] Miek Gieben. *Go DNS package*. https://miek.nl/2014/august/16/go-dns-package/. (Accessed on 04/17/2024). Aug. 2014.

[134] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. “Algorand: Scaling byzantine agreements for cryptocurrencies”. In: *Proceedings of the 26th symposium on operating systems principles*. 2017, pp. 51–68.

[135] Sharon Goldberg, Leonid Reyzin, Dimitrios Papadopoulos, and Jan Včelák. *Verifiable Random Functions (VRFs)*. RFC 9381. Aug. 2023.
DOI: 10.17487/RFC9381.

[136] Google. *DNS-over-HTTPS (DoH) | Public DNS | Google Developers*. https://developers.google.com/speed/public-dns/docs/doh. (Accessed on 09/15/2020).

[137] Google. *IP ranges that Google makes available to users on the internet*. https://www.gstatic.com/ipranges/goog.json. (Accessed on 04/18/2024). Apr. 2024.

[138] United States Dot Gov. *Recent Updates*. https://home.dotgov.gov/. (Accessed on 05/25/2020). Mar. 2020.

[139] John Graham-Cumming. *Announcing the Results of the 1.1.1.1 Public DNS Resolver Privacy Examination*. https://blog.cloudflare.com/announcing-the-results-of-the-1-1-1-1-public-dns-resolver-privacy-examination/. (Accessed on 09/15/2020). Mar. 2020.

[140] Benjamin Greschbach, Tobias Pulls, Laura M Roberts, Philipp Winter, and Nick Feamster. "The effect of DNS on Tor's anonymity". In: *arXiv preprint arXiv:1609.08187* (2016).

[141] Ilya Grigorik. "Making the web faster with HTTP 2.0". In: *Communications of the ACM* 56.12 (2013), pp. 42–49.

[142] GSM Association and others. "The mobile economy 2022". In: *https://www.gsmaintelligence.com/research* (2022).

[143] Alexander G Haddad. "Effects of Internet Exclusion on the City of Detroit". In: *Honors College Thesis* (2022).
URL: https://digitalcommons.wayne.edu/honorstheses/77.

[144] Shaddi Hasan, Mary Claire Barela, Matthew Johnson, Eric Brewer, and Kurtis Heimerl. "Scaling Community Cellular Networks with CommunityCellularManager". In: *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*. Boston, MA: USENIX Association, 2019, pp. 735–750. ISBN: 978-1-931971-49-2.
ISBN: 978-1-931971-49-2.

[145] Shaddi Hasan, Kurtis Heimerl, Kate Harrison, Kashif Ali, Sean Roberts, Anant Sahai, and Eric Brewer. "GSM whitespaces: An opportunity for rural cellular service". In: *2014 IEEE International Symposium on Dynamic Spectrum Access Networks (DYSPAN)*. IEEE. 2014, pp. 271–282.

[146] Kurtis Heimerl and Eric Brewer. "The village base station". In: *Proceedings of the 4th ACM Workshop on Networked Systems for Developing Regions (NSDR)*. ACM. 2010, p. 14.

[147] Kurtis Heimerl, Shaddi Hasan, Kashif Ali, Eric Brewer, and Tapan Parikh. "Local, sustainable, small-scale cellular networks". In: *Proceedings of the Sixth International Conference on Information and Communication Technologies and Development (ICTD)*. ACM. 2013, pp. 2–12.

[148] Kurtis Heimerl, Shaddi Hasan, Kashif Ali, Tapan Parikh, and Eric Brewer. "A longitudinal study of local, sustainable, small-scale cellular networks". In: *Information Technologies & International Development* 11.1 (2015), pp–1.

[149] Martin Hellebrandt and Rudolf Mathar. "Location tracking of mobiles in cellular radio networks". In: *IEEE transactions on Vehicular Technology* 48.5 (1999), pp. 1558–1562.

[150] Nadia Heninger, Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. "Mining your Ps and Qs: detection of widespread weak keys in network devices". In: *Proceedings of the 21st USENIX Conference on Security Symposium*. Security'12. Bellevue, WA: USENIX Association, 2012, p. 35.

[151] Ansel Herz. *Judge Who Authorized Police Search of Seattle Privacy Activists Wasn't Told They Operate Tor Network*. https://web.archive.org/web/20191210114929/https://www.thestranger.com/slog/2016/04/08/23914735/judge-who-authorized-police-search-of-seattle-privacy-activists-wasnt-told-they-operate-tor-network/. (Accessed on 09/15/2020). Apr. 2016.

[152] Amir Herzberg and Haya Shulman. "Towards adoption of dnssec: Availability and security challenges". In: *Cryptology ePrint Archive* (2013).

[153] Amir Herzberg and Haya Shulman. "Vulnerable Delegation of DNS Resolution". In: *Computer Security – ESORICS 2013*. Ed. by Jason Crampton, Sushil Jajodia, and Keith Mayes. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 219–236. ISBN: 978-3-642-40203-6. ISBN: 978-3-642-40203-6.

[154] Rae Hodge. *India Orders VPN Companies to Collect and Hand Over User Data*. https://www.cnet.com/news/privacy/india-orders-vpn-companies-to-collect-and-hand-over-user-data/. May 2022.

[155] Bjoern Hoehrmann. *Scripting Media Types*. RFC 4329. Apr. 2006. DOI: 10.17487/RFC4329.

[156] Paul Hoffman and Patrick McManus. "DNS queries over HTTPS (DoH)". In: *Internet Requests for Comments, IETF, RFC* 8484 (2018).

[157] Pi Hole. *Pi-hole - A black hole for Internet advertisements*. https://pi-hole.net/. (Accessed on 09/16/2020).

[158] Austin Hounsel, Paul Schmitt, Kevin Borgolte, and Nick Feamster. *Measuring the Performance of Encrypted DNS Protocols from Broadband Access Networks*. 2020.

[159] Russ Housley. *Guidelines for Cryptographic Algorithm Agility and Selecting Mandatory-to-Implement Algorithms*. RFC 7696. Nov. 2015. DOI: 10.17487/RFC7696.

[160] Zi Hu, Liang Zhu, John Heidemann, Allison Mankin, Duane Wessels, and Paul Hoffman. "Specification for DNS over transport layer security (TLS)". In: *IETF RFC7858, May* (2016).

[161] Mei Lin Hui and Gavin Lowe. “Fault-preserving simplifying transformations for security protocols”. In: *Journal of Computer Security* 9.1-2 (2001), pp. 3–46.

[162] Nathaniel Husted and Steven Myers. “Mobile location tracking in metro areas: malnets and others”. In: *Proceedings of the 17th ACM conference on Computer and communications security*. 2010, pp. 85–96.

[163] The Linux Foundation Hyperledger Sawtooth. *PoET 1.0 Specification*. https://sawtooth.hyperledger.org/docs/core/releases/1.0/architecture/poet.html. Accessed: 2019-11-4.

[164] ICANN. *Country code top-level domain - ICANNWiki*. https://icannwiki.org/Country_code_top-level_domain. (Accessed on 03/06/2020). Apr. 2017.

[165] European Telecommunications Standards Institute. *Digital cellular telecommunications systems (Phase 2+); Lawful Interception - Stage 1 (GSM 02.33) Version 8.0.1*. en. Jan. 1997. URL: https://www.etsi.org/deliver/etsi_gts/02/0233/05.00.00_60/gsmts_0233v050000p.pdf.

[166] Internet Assigned Numbers Authority (IANA). *IANA Root Anchors Data*. Accessed: November 30, 2023. 2017.
URL: https://data.iana.org/root-anchors/root-anchors.xml.

[167] Sergi Isasi and Vicky Shrestha. *Expanding DNSSEC Adoption*. https://blog.cloudflare.com/automatically-provision-and-maintain-dnssec. (Accessed on 04/16/2024). Sept. 2018.

[168] Johnathan Ishmael, Sara Bury, Dimitrios Pezaros, and Nicholas Race. “Deploying rural community wireless mesh networks”. In: *IEEE Internet Computing* 12.4 (2008), pp. 22–29.

[169] *Issue 116982: Chromium opens useless TCP connections*. Chromium Bugs. Mar. 2012.
URL: https://bugs.chromium.org/p/chromium/issues/detail?id=116982#c6.

[170] Esther Jang, Mary Claire Barela, Matt Johnson, Philip Martinez, Cedric Festin, Margaret Lynn, Josephine Dionisio, and Kurtis Heimerl. “Crowdsourcing Rural Network Maintenance and Repair via Network Messaging”. In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI ’18. Montreal QC, Canada: Association for Computing Machinery, 2018, pp. 1–12.
DOI: 10.1145/3173574.3173641.

[171] Esther Han Beol Jang, Philip Garrison, Ronel Vincent Vistal, Maria Theresa D. Cunanan, Maria Theresa Perez, Philip Martinez, Matthew William Johnson, John Andrew Evangelista, Syed Ishtiaque Ahmed, Josephine Dionisio, Mary Claire Aguilar Barela, and Kurtis Heimerl. “Trust and Technology Repair Infrastructures in the Remote Rural Philippines: Navigating Urban-Rural Seams”. In: *Proc. ACM Hum.-Comput. Interact.* 3.CSCW (Nov. 2019).
DOI: 10.1145/3359201.

[172] Lin Jin, Shuai Hao, Haining Wang, and Chase Cotton. “Understanding the impact of encrypted DNS on internet censorship”. In: *Proceedings of the Web Conference 2021*. 2021, pp. 484–495.

[173] Matthew Johnson, Spencer Sevilla, Esther Jang, and Kurtis Heimerl. "dLTE: Building a More WiFi-like Cellular Network: (Instead of the Other Way Around)". In: *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*. HotNets '18. Redmond, WA, USA: ACM, 2018, pp. 8–14.
DOI: 10.1145/3286062.3286064.

[174] Matthew Johnson, Sudheesh Singanamalla, Nick Durand, Esther Jang, Spencer Sevilla, and Kurtis Heimerl. "dAuth: A Resilient Authentication Architecture for Federated Private Cellular Networks". In: *Proceedings of the 2024 ACM SIGCOMM 2024 Conference*. SIGCOMM '24. Sydney, Australia: Association for Computing Machinery, 2024.

[175] Matthew William Johnson, Esther Han Beol Jang, Frankie O'Rourke, Rachel Ye, and Kurtis Heimerl. "Network Capacity as Common Pool Resource: Community-Based Congestion Management in a Community Network". In: *Proc. ACM Hum.-Comput. Interact.* 5.CSCW1 (Apr. 2021).
DOI: 10.1145/3449135.

[176] Dixon Jones. *Majestic Million CSV now free for all, daily.* (Accessed on 09/07/2020). Oct. 2012.

[177] Roger Piqueras Jover and Joshua Lackey. "dHSS-distributed Peer-to-Peer implementation of the LTE HSS based on the bitcoin/namecoin architecture". In: *2016 IEEE International Conference on Communications Workshops (ICC)*. IEEE. 2016, pp. 354–359.

[178] Georgios Kambourakis, Tassos Moschos, Dimitris Geneiatakis, and Stefanos Gritzalis. "Detecting DNS amplification attacks". In: *Critical Information Infrastructures Security: Second International Workshop, CRITIS 2007, Málaga, Spain, October 3-5, 2007. Revised Papers 2*. Springer. 2008, pp. 185–196.

[179] Dan Kaminsky. *Black Ops 2008: It's The End Of The Cache As We Know It*. en. 2008.

[180] Ju-Sung Kang, Sang-Uk Shin, Dowon Hong, and Okyeon Yi. "Provable security of KASUMI and 3GPP encryption mode f 8". In: *Advances in Cryptology-ASIACRYPT 2001: 7th International Conference on the Theory and Application of Cryptology and Information Security Gold Coast, Australia, December 9–13, 2001 Proceedings 7*. Springer. 2001, pp. 255–271.

[181] Jonathan Katz. *Digital signatures*. Vol. 1. Springer, 2010.

[182] Niall Keleher, Mary Claire Barela, Joshua Blumenstock, Cedric Festin, Matthew Podolsky, Erin Troland, Arman Rezaee, and Kurtis Heimerl. "Connecting Isolated Communities: Quantitative Evidence on the Adoption of Community Cellular Networks in the Philippines". In: *Proceedings of the 2020 International Conference on Information and Communication Technologies and Development*. ICTD '20. Guayaquil, Ecuador: Association for Computing Machinery, 2020.
DOI: 10.1145/3392561.3394645.

[183] Franziskus Kiefer. *Improving AES-GCM Performance - Mozilla Security Blog*. https://blog.mozilla.org/security/2017/09/29/improving-aes-gcm-performance/. (Accessed on 09/16/2020). Sept. 2017.

[184] Zak Killian. *Five-Year Syniverse Data Breach May Have Leaked Trillions Of Mobile User Text Messages*. https://hothardware.com/news/syniverse-breach-leak-text-messages. (Accessed on 01/18/2022). Oct. 2021.

[185] Eric Kinnear, Patrick McManus, Tommy Pauly, and Christopher Wood. *Oblivious DNS Over HTTPS–IETF Draft*. https://tools.ietf.org/html/draft-pauly-dprive-oblivious-doh-01. 2019.

[186] Scott Kitterman. *Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1*. RFC 7208. Apr. 2014.
DOI: 10.17487/RFC7208.

[187] Aniket Kittur, Ed H. Chi, and Bongwon Suh. "Crowdsourcing User Studies with Mechanical Turk". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '08. Florence, Italy: Association for Computing Machinery, 2008, pp. 453–456.
DOI: 10.1145/1357054.1357127.

[188] Amit Klein, Haya Shulman, and Michael Waidner. "Internet-wide study of DNS cache injections". In: *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*. Atlanta, GA, USA: IEEE, 2017, pp. 1–9.
DOI: 10.1109/INFOCOM.2017.8057202.

[189] Erik Kline. *DNS over TLS support in Android P Developer Preview*. https://android-developers.googleblog.com/2018/04/dns-over-tls-support-in-android-p.html. (Accessed on 09/15/2020). Apr. 2018.

[190] Ronny Ko, James Mickens, Blake Loring, and Ravi Netravali. "Oblique: Accelerating Page Loads Using Symbolic Execution". In: *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. USENIX Association, Apr. 2021, pp. 289–302. ISBN: 978-1-939133-21-2.
ISBN: 978-1-939133-21-2.

[191] Geir M Køien. "Mutual entity authentication for LTE". In: *2011 7th International Wireless Communications and Mobile Computing Conference*. IEEE. 2011, pp. 689–694.

[192] Brian Krebs. *It's Way Too Easy to Get a .gov Domain Name - Krebs on Security*. https://krebsonsecurity.com/2019/11/its-way-too-easy-to-get-a-gov-domain-name/. (Accessed on 05/25/2020). Nov. 2019.

[193] Balachander Krishnamurthy and Craig E Wills. "Analyzing factors that influence end-to-end web performance". In: *Computer Networks* 33.1-6 (2000), pp. 17–32.

[194] Murray Kucherawy, Dave Crocker, and Tony Hansen. *DomainKeys Identified Mail (DKIM) Signatures*. RFC 6376. Sept. 2011.
DOI: 10.17487/RFC6376.

[195] Murray Kucherawy and Elizabeth Zwicky. *Domain-based Message Authentication, Reporting, and Conformance (DMARC)*. RFC 7489. Mar. 2015.
DOI: 10.17487/RFC7489.

[196] Marc Kührer, Thomas Hupperich, Jonas Bushart, Christian Rossow, and Thorsten Holz. "Going wild: Large-scale classification of open DNS resolvers". In: *Proceedings of the 2015 Internet Measurement Conference*. 2015, pp. 355–368.

[197] Marc Kührer, Thomas Hupperich, Christian Rossow, and Thorsten Holz. "Exit from hell? Reducing the impact of amplification DDoS attacks". In: *23rd {USENIX} Security Symposium ({USENIX} Security 14)*. 2014, pp. 111–125.

[198] NLNet Labs. *Unbound - Howto enable DNSSEC*. https://nlnetlabs.nl/documentation/unbound/howto-anchor/. (Accessed on 11/26/2023). 2023.

[199] Evan Lam, Richard Anderson, Kurtis Heimerl, Yurie Ito, Jonathan Joseph de Koning, Adam M Lange, Jarrod O'Malley, Adam Shostack, Arastoo Taslim, and Sudheesh Singanamalla. "Poster: On the (In) Security of Government Web and Mail Infrastructure". In: NDSS. 2024.

[200] Ulf Lamping and Ed Warnicke. "Wireshark user's guide". In: *Interface* 4.6 (2004), p. 1.

[201] Adam Langley. *Serializing DNS Records with DNSSEC Authentication*. Internet-Draft draft-agl-dane-serializechain-01. Work in Progress. Internet Engineering Task Force, July 2011. 20 pp.
URL: https://datatracker.ietf.org/doc/draft-agl-dane-serializechain/01/.

[202] Ben Laurie. "Certificate transparency". In: *Communications of the ACM* 57.10 (2014), pp. 40–46.

[203] Ben Laurie, Adam Langley, and Emilia Kasper. *Certificate Transparency*. RFC 6962. June 2013.
DOI: 10.17487/RFC6962.

[204] Ben Laurie, Adam Langley, Emilia Kasper, Eran Messeri, and Rob Stradling. *Certificate Transparency Version 2.0*. RFC 9162. Dec. 2021.
DOI: 10.17487/RFC9162.

[205] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. "Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation". In: *Proceedings of the 26th Annual Network and Distributed System Security Symposium*. NDSS 2019. Feb. 2019.
DOI: 10.14722/ndss.2019.23386.

[206] Microsoft Learn. *DNS Clients | Security-aware client*. https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn593685(v=ws.11)#security-aware-client. (Accessed on 11/26/2023). Aug. 2016.

[207] Brandon LeBlanc. *Announcing Windows 10 Insider Preview Build 20185*. https://blogs.windows.com/windows-insider/2020/08/05/announcing-windows-10-insider-preview-build-20185/. (Accessed on 09/15/2020). Aug. 2020.

[208] Julia Len, Esha Ghosh, Paul Grubbs, and Paul Rösler. "Interoperability in End-to-End Encrypted Messaging". In: *Cryptology ePrint Archive* (2023).

[209] Matt Lepinski and Kotikalapudi Sriram. *BGPsec Protocol Specification*. RFC 8205. Sept. 2017. DOI: 10.17487/RFC8205.

[210] Bingyu Li, Jingqiang Lin, Fengjun Li, Qiongxiao Wang, Qi Li, Jiwu Jing, and Congli Wang. "Certificate Transparency in the Wild: Exploring the Reliability of Monitors". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. CCS '19. London, United Kingdom: Association for Computing Machinery, 2019. ISBN: 9781450367479.
ISBN: 9781450367479.

[211] Wilson Lian, Eric Rescorla, Hovav Shacham, and Stefan Savage. "Measuring the practical impact of DNSSEC deployment". In: *22nd USENIX Security Symposium (USENIX Security 13)*. 2013, pp. 573–588.

[212] Arch Linux. *systemd-resolved - ArchWiki*. https://wiki.archlinux.org/title/systemd-resolved. (Accessed on 11/26/2023). Nov. 2023.

[213] Chaoyi Lu, Baojun Liu, Zhou Li, Shuang Hao, Haixin Duan, Mingming Zhang, Chunying Leng, Ying Liu, Zaifeng Zhang, and Jianping Wu. "An End-to-End, Large-Scale Measurement of DNS-over-Encryption: How Far Have We Come?" In: *Proceedings of the Internet Measurement Conference*. 2019, pp. 22–35.

[214] Zhihong Luo, Silvery Fu, Mark Theis, Shaddi Hasan, Sylvia Ratnasamy, and Scott Shenker. "Democratizing Cellular Access with CellBricks". In: *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*. SIGCOMM '21. Virtual Event, USA: Association for Computing Machinery, 2021, pp. 626–640.
DOI: 10.1145/3452296.3473336.

[215] Doug Madory. *Digging into the Orange España Hack*. https://www.kentik.com/blog/digging-into-the-orange-espana-hack/. Jan. 2024.

[216] Harjasleen Malvai, Lefteris Kokoris-Kogias, Alberto Sonnino, Esha Ghosh, Ercan Oztürk, Kevin Lewi, and Sean Lawlor. "Parakeet: Practical key transparency for end-to-end encrypted messaging". In: *Cryptology ePrint Archive* (2023).

[217] Priyatosh Mandal. "PCRF: On the blocking probability of LTE-A". In: *2020 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. IEEE. 2020, pp. 1–4.

[218] Electronic Frontier Foundation Marcia Hoffmann. *Why IP Addresses Alone Don't Identify Criminals*. https://www.eff.org/deeplinks/2011/08/why-ip-addresses-alone-dont-identify-criminals. (Accessed on 09/15/2020). Aug. 2011.

[219] Moxie Marlinspike and Trevor Perrin. "The X3DH key agreement protocol". In: *Open Whisper Systems* 283.10 (2016).

[220] Andrés Martínez Fernández, José Vidal Manzano, Javier Simó Reigadas, Ignacio Prieto Egido, Adrián Agustín de Dios, Juan Paco, and Álvaro Rendón. "The TUCAN3G project: wireless technologies for isolated rural communities in developing countries based on 3G small-cell deployments". In: *IEEE communications magazine* 54.7 (2016), pp. 36–43.

[221] Maria Massaro and Fernando Beltrán. "Will 5G lead to more spectrum sharing? Discussing recent developments of the LSA and the CBRS spectrum sharing frameworks". In: *Telecommunications Policy* 44.7 (2020), p. 101973.

[222] Stephanos Matsumoto and Raphael M Reischuk. "Certificates-as-an-Insurance: Incentivizing accountability in SSL/TLS". In: *Proceedings of the NDSS Workshop on Security of Emerging Network Technologies (SENT'15)*. 2015.

[223] Erika McCallister, Tim Grance, and Karen Scarfone. *Guide to protecting the confidentiality of Personally Identifiable Information (PII): Recommendations of the National Institute of Standards and Technology*. NIST special publication ; 800-122. Computer security. Gaithersburg, MD: U.S. Dept. of Commerce, National Institute of Standards and Technology, 2010.

[224] Brendan McMillion. *End-to-End Integrity with IPFS*. https://blog.cloudflare.com/e2e-integrity/. (Accessed on 02/13/2023). Sept. 2018.

[225] Patrick Meenan and Contributors. *Web Page Test - Web Performance Testing Code*. https://github.com/WPO-Foundation/webpagetest. 2021.

[226] Marcela S Melara, Aaron Blankstein, Joseph Bonneau, Edward W Felten, and Michael J Freedman. "CONIKS: Bringing Key Transparency to End Users". In: *24th USENIX Security Symposium (USENIX Security 15)*. 2015, pp. 383–398.

[227] NYC Mesh. *NYC Mesh*. 2020.

[228] Silvio Micali, Michael Rabin, and Salil Vadhan. "Verifiable random functions". In: *40th annual symposium on foundations of computer science (cat. No. 99CB37039)*. IEEE. 1999, pp. 120–130.

[229] Microsoft. *List of Participants - Microsoft Trusted Root Program*. https://docs.microsoft.com/en-us/security/trusted-root/participants-list. (Accessed on 09/07/2020. Aug. 2019.

[230] Ariana Mirian, Christopher Thompson, Stefan Savage, Geoffrey M Voelker, and Adrienne Porter Felt. "HTTPS Adoption in the Longtail". In: (2018).

[231] Paul Mockapetris. *Domain names - concepts and facilities*. RFC 1034. Nov. 1987. DOI: 10.17487/RFC1034.

[232] Paul Mockapetris. *Domain names - implementation and specification*. RFC 1035. Nov. 1987. DOI: 10.17487/RFC1035.

[233] Paul Mockapetris and Kevin J Dunlap. "Development of the domain name system". In: *Symposium proceedings on Communications architectures and protocols*. 1988, pp. 123–133.

[234] Jeffrey Mogul. *RFC 919: Broadcasting Internet Datagrams*. 1984.

[235] Bodo Möller, Thai Duong, and Krzysztof Kotowicz. “This POODLE bites: exploiting the SSL 3.0 fallback”. In: *Security Advisory* (2014).

[236] Giovane C. M. Moura, Sebastian Castro, John Heidemann, and Wes Hardaker. “TsuNAME: Exploiting Misconfiguration and Vulnerability to DDoS DNS”. In: *Proceedings of the 21st ACM Internet Measurement Conference*. IMC ’21. Virtual Event: Association for Computing Machinery, 2021, pp. 398–418.
DOI: 10.1145/3487552.3487824.

[237] Giovane C.M. Moura, Sebastian Castro, Wes Hardaker, Maarten Wullink, and Cristian Hesselman. “Clouding up the internet: How centralized is dns traffic becoming?” In: *Proceedings of the ACM Internet Measurement Conference*. 2020, pp. 42–49.

[238] Mozilla. *Cert Verifier - Extended Validation*. https://hg.mozilla.org/mozilla-central/file/tip/security/certverifier/ExtendedValidation.cpp. (Accessed on 03/04/2020). 2020.

[239] Mozilla. *Comcast’s Xfinity Internet Service Joins Firefox’s Trusted Recursive Resolver Program - The Mozilla Blog*. https://blog.mozilla.org/blog/2020/06/25/comcasts-xfinity-internet-service-joins-firefoxs-trusted-recursive-resolver-program/. (Accessed on 09/15/2020). June 2020.

[240] Mozilla. *Mozilla Included CA Certificate List*. https://wiki.mozilla.org/CA/Included_Certificates. (Accessed on 09/07/2020). Oct. 2017.

[241] Mozilla. *Mozilla Policy Requirements for DNS over HTTPs Partners*. https://wiki.mozilla.org/Security/DOH-resolver-policy. (Accessed on 09/15/2020). Sept. 2020.

[242] Mozilla. *Mozilla Root Store Policy*. https://www.mozilla.org/en-US/about/governance/policies/security-group/certs/policy/. (Accessed on 05/07/2024). Sept. 2023.

[243] Alec Muffet. “No Port 53, Who Dis?; A Year of DNS over HTTPS over Tor”. In: *NDSS DNS Privacy Workshop*. Feb. 2021.

[244] Alec Muffett. *DoHoT: making practical use of DNS over HTTPS over Tor*. https://github.com/alecmuffett/dohot. (Accessed on 09/15/2020). July 2020.

[245] Satoshi Nakamoto et al. “Bitcoin: A peer-to-peer electronic cash system”. In: (2008).

[246] David Naylor, Alessandro Finamore, Ilias Leontiadis, Yan Grunenberger, Marco Mellia, Maurizio Munafò, Konstantina Papagiannaki, and Peter Steenkiste. “The cost of the" s" in https”. In: *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. 2014, pp. 133–140.

[247] Arian Akhavan Niaki, Nguyen Phong Hoang, Phillipa Gill, Amir Houmansadr, et al. “Triplet Censors: Demystifying Great {Firewall’s}{DNS} Censorship Behavior”. In: *10th USENIX Workshop on Free and Open Communications on the Internet (FOCI 20)*. 2020.

[248] Henrik Frystyk Nielsen, James Gettys, Anselm Baird-Smith, Eric Prud’hommeaux, Håkon Wium Lie, and Chris Lilley. “Network performance effects of HTTP/1.1, CSS1, and PNG”. In: *Proceedings of the ACM SIGCOMM’97 conference on Applications, technologies, architectures, and protocols for computer communication*. 1997, pp. 155–166.

[249] NIST. *NVD - CVE-2013-2566*. https://nvd.nist.gov/vuln/detail/CVE-2013-2566. (Accessed on 09/16/2020). Mar. 2013.

[250] Karsten Nohl and Sascha Krißler. "Subverting the security base of GSM". In: *Hacking at Random* (2009).

[251] Mark Nottingham and Erik Nygren. *The ORIGIN HTTP/2 Frame*. RFC 8336. Mar. 2018. DOI: 10.17487/RFC8336.

[252] *ODoH Artifacts*. https://github.com/sudheesh001/ODoH-Artifacts.

[253] Jan Odvarko. *HAR 1.2 Spec*. http://www.softwareishard.com/blog/har-12-spec/. (Accessed on 02/28/2021).

[254] Kelly Olson, Mic Bowman, James Mitchell, Shawn Amundson, Dan Middleton, and Cian Montgomery. "Sawtooth: An Introduction". In: *The Linux Foundation, Jan* (2018).

[255] Magnus Olsson, Catherine Mulligan, Shabnam Sultana, Stefan Rommer, and Lars Frid. *EPC and 4G packet networks: driving the mobile broadband revolution*. Academic Press, 2012.

[256] Shree Om, Carlos Rey-Moreno, and William David Tucker. "Towards a scalability model for wireless mesh networks". In: *Technical Report* (2015).

[257] Open5Gs. *NextEPC/Open5Gs: C Implementation of the 3GPP LTE EPC*. https://github.com/open5gs/open5gs. 2019.

[258] OpenSSL. *OpenSSL: Cryptography and SSL/TLS toolkit*. https://www.openssl.org/. (Accessed on 03/06/2020).

[259] OpenSSL. *Verify - OpenSSL*. https://www.openssl.org/docs/man1.0.2/man1/verify.html. (Accessed on 09/21/2020).

[260] Gustaf Ouvrier, Michel Laterman, Martin Arlitt, and Niklas Carlsson. "Characterizing the HTTPS trust landscape: a passive view from the edge". In: *IEEE Communications Magazine* 55.7 (2017), pp. 36–42.

[261] Martine Paris. *Google and Mozilla block Kazakhstan root CA certificate from Chrome and Firefox*. https://venturebeat.com/security/google-and-mozilla-block-kazakhstan-root-ca-certificate-from-chrome-and-firefox/. Aug. 2019.

[262] Jeman Park, Aminollah Khormali, Manar Mohaisen, and Aziz Mohaisen. "Where are you taking me? Behavioral analysis of open DNS resolvers". In: *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE. 2019, pp. 493–504.

[263] Shinjo Park. "Why we cannot win: on fake base stations and their detection methods". PhD thesis. Technische Universität Berlin, 2023.

[264] Rabin K Patra, Sergiu Nedevschi, Sonesh Surana, Anmol Sheth, Lakshminarayanan Subramanian, and Eric A Brewer. "WiLDNet: Design and Implementation of High Performance WiFi Based Long Distance Networks." In: *NSDI*. Vol. 1. 2007, p. 1.

[265] Paul Pearce, Ben Jones, Frank Li, Roya Ensafi, Nick Feamster, Nick Weaver, and Vern Paxson. "Global measurement of DNS manipulation". In: *26th USENIX Security Symposium*. 2017.

[266] Trevor Perrin and Moxie Marlinspike. "The double ratchet algorithm". In: *GitHub wiki* 112 (2016).

[267] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. "Tranco: A research-oriented top sites ranking hardened against manipulation". In: *Network and Distributed Systems Security (NDSS) Symposium* (2019).

[268] EU Resolver Policy. *European DNS Resolver Policy*. en-GB. 2021. URL: https://europeanresolverpolicy.com/.

[269] Matthew Prince. *Introducing 1.1.1.1 for Families*. https://blog.cloudflare.com/introducing-1-1-1-1-for-families/. 2020.

[270] J Ronald Prins and Business Unit Cybercrime. "Diginotar certificate authority breach "operation black tulip"". In: *Fox-IT, November* 18 (2011).

[271] Chromium Projects. *DNS over HTTPS (aka DoH)*. https://www.chromium.org/developers/dns-over-https. (Accessed on 09/15/2020).

[272] DNSCrypt Proxy. *Anonymized DNS Wiki*. https://github.com/DNSCrypt/dnscrypt-proxy/wiki/Anonymized-DNS. (Accessed on 09/15/2020).

[273] FIPS PUB. "Security Requirements for Cryptographic Modules". In: *FIPS PUB* 140 (1994).

[274] Jeremy Quirke. "Security in the GSM system". In: *AusMobile, May* (2004), pp. 1–26.

[275] Steve Ragan. *Trustwave Explains Subordinate Certificate Logic - SecurityWeek*. https://www.securityweek.com/trustwave-explains-subordinate-certificate-logic/. Feb. 2012.

[276] Ram Sundara Raman, Leonid Evdokimov, Eric Wurstrow, J. Alex Halderman, and Roya Ensafi. "Investigating Large Scale HTTPS Interception in Kazakhstan". In: *Proceedings of the ACM Internet Measurement Conference*. IMC '20. Virtual Event, USA: Association for Computing Machinery, 2020, pp. 125–132. DOI: 10.1145/3419394.3423665.

[277] Reddit Communities. *DNS query average : PiHole*. https://www.reddit.com/r/pihole/comments/a8ngnu/dns_query_average/. (Accessed on 09/15/2020). Dec. 2018.

[278] "Regulation (EU) 2022/1925 of the European Parliament and of the Council of 14 September 2022 on contestable and fair markets in the digital sector and amending Directives (EU) 2019/1937 and (EU) 2020/1828 (Digital Markets Act)". In: *OJ* L 265 (Oct. 2022), pp. 1–66.

[279] Charles Reis, Alexander Moshchuk, and Nasko Oskov. "Site isolation: process separation for web sites within the browser". In: *28th USENIX Security Symposium (USENIX Security 19)*. 2019, pp. 1661–1678.

[280] Yakov Rekhter, B Moskowitz, Daniel Karrenberg, GJ de Groot, and Eliot Lear. *Rfc1918: Address allocation for private internets*. 1996.

[281] GFW Report. *I'll shake your hand: what happens after DNS poisoning | Issue #294 | net4people/bbs*. https://github.com/net4people/bbs/issues/294. (Accessed on 10/11/2023). Sept. 2023.

[282] Eric Rescorla and Tim Dierks. *The Transport Layer Security (TLS) Protocol Version 1.2*. RFC 5246. Aug. 2008.
DOI: 10.17487/RFC5246.

[283] Eric Rescorla, Kazuho Oku, Nick Sullivan, and Christopher A. Wood. *TLS Encrypted Client Hello*. Internet-Draft draft-ietf-tls-esni-17. Work in Progress. Internet Engineering Task Force, Oct. 2023. 48 pp.
URL: https://datatracker.ietf.org/doc/draft-ietf-tls-esni/17/.

[284] *Rhizomatica*. https://www.rhizomatica.org.

[285] Roland van Rijswijk-Deij, Anna Sperotto, and Aiko Pras. "DNSSEC and its potential for DDoS attacks: a comprehensive measurement study". In: *Proceedings of the 2014 Conference on Internet Measurement Conference*. 2014, pp. 449–460.

[286] Scott Rose, Matt Larson, Dan Massey, Rob Austein, and Roy Arends. *DNS Security Introduction and Requirements*. RFC 4033. Mar. 2005.
DOI: 10.17487/RFC4033.

[287] Scott Rose, Matt Larson, Dan Massey, Rob Austein, and Roy Arends. *Protocol Modifications for the DNS Security Extensions*. RFC 4035. Mar. 2005.
DOI: 10.17487/RFC4035.

[288] Scott Rose, Matt Larson, Dan Massey, Rob Austein, and Roy Arends. *Resource Records for the DNS Security Extensions*. RFC 4034. Mar. 2005.
DOI: 10.17487/RFC4034.

[289] S2W. *Post Mortem of KlaySwap Incident through BGP Hijacking*. https://medium.com/s2wblog/post-mortem-of-klayswap-incident-through-bgp-hijacking-en-3ed7e33de600. Feb. 2022.

[290] Barsha Saha, Keya Sengupta, Rohit Joshi, and Sharad Nath Bhattacharya. "Evolution of Competition in Telecom Oligopoly–A Systematic Analysis in Post-Privatization Era." In: *International Journal of Business & Economics* 21.2 (2022).

[291] Constantin Sander, Leo Blöcher, Klaus Wehrle, and Jan Rüth. "Sharding and HTTP/2 Connection Reuse Revisited: Why Are There Still Redundant Connections?" In: *Proceedings of the 21st ACM Internet Measurement Conference*. IMC '21. Virtual Event: Association for Computing Machinery, 2021, pp. 292–301.

[292] Stefan Santesson, Michael Myers, Rich Ankney, Ambarish Malpani, Slava Galperin, and Dr. Carlisle Adams. *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP*. RFC 6960. June 2013.
DOI: 10.17487/RFC6960.

[293] Sambhav Satija and Rahul Chatterjee. "BlindTLS: Circumventing TLS-based HTTPS censorship". In: *Proceedings of the ACM SIGCOMM 2021 Workshop on Free and Open Communications on the Internet*. FOCI '21. Virtual Event, USA: Association for Computing Machinery, 2021, pp. 43–49.
DOI: 10.1145/3473604.3474564.

[294] Sambhav Satija, Apurv Mehra, Sudheesh Singanamalla, Karan Grover, Muthian Sivathanu, Nishanth Chandran, Divya Gupta, and Satya Lokam. "Blockene: A high-throughput blockchain over mobile devices". In: *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*. 2020, pp. 567–582.

[295] Kazunori Sato. "An inside look at google bigquery". In: *White paper, URL: https://cloud. google. com/files/BigQueryTechnicalWP. pdf* (2012).

[296] Quirin Scheitle, Oliver Gasser, Theodor Nolte, Johanna Amann, Lexi Brent, Georg Carle, Ralph Holz, Thomas C Schmidt, and Matthias Wählisch. "The rise of certificate transparency and its implications on the internet ecosystem". In: *Proceedings of the Internet Measurement Conference 2018*. 2018, pp. 343–349.

[297] Jeffrey I. Schiller. *Strong Security Requirements for Internet Engineering Task Force Standard Protocols*. RFC 3365. Aug. 2002.
DOI: 10.17487/RFC3365.

[298] David Schinazi and Tommy Pauly. *Happy Eyeballs Version 2: Better Connectivity Using Concurrency*. RFC 8305. Dec. 2017.
DOI: 10.17487/RFC8305.

[299] Brandon Schlinker, Italo Cunha, Yi-Ching Chiu, Srikanth Sundaresan, and Ethan Katz-Bassett. "Internet performance from facebook's edge". In: *Proceedings of the Internet Measurement Conference*. 2019, pp. 179–194.

[300] Benedikt Schmidt, Simon Meier, Cas Cremers, and David Basin. "Automated Analysis of Diffie-Hellman Protocols and Advanced Security Properties". In: *25th IEEE Computer Security Foundations Symposium, CSF 2012, Cambridge, MA, USA, June 25-27, 2012*. Ed. by Stephen Chong. IEEE, 2012, pp. 78–94.

[301] Paul Schmitt, Anne Edmundson, Allison Mankin, and Nick Feamster. "Oblivious DNS: Practical Privacy for DNS Queries: Published in PoPETS 2019". In: *Proceedings of the Applied Networking Research Workshop*. ANRW '19. Montreal, Quebec, Canada: Association for Computing Machinery, 2019, pp. 17–19.
DOI: 10.1145/3340301.3341128.

[302] Paul Schmitt and Barath Raghavan. "Pretty Good Phone Privacy". In: *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, Aug. 2021, pp. 1737–1754. ISBN: 978-1-939133-24-3.
ISBN: 978-1-939133-24-3.

[303] Kyle Schomp, Tom Callahan, Michael Rabinovich, and Mark Allman. "Assessing DNS Vulnerability to Record Injection". In: *Passive and Active Measurement*. Ed. by Michalis Faloutsos and Aleksandar Kuzmanovic. Cham: Springer International Publishing, 2014, pp. 214–223. ISBN: 978-3-319-04918-2.
ISBN: 978-3-319-04918-2.

[304] Benjamin M. Schwartz, Mike Bishop, and Erik Nygren. *Service binding and parameter specification via the DNS (DNS SVCB and HTTPS RRs)*. Internet-Draft draft-ietf-dnsop-svcb-https-03. Work in Progress. Internet Engineering Task Force, Feb. 2021. 47 pp.
URL: https://datatracker.ietf.org/doc/html/draft-ietf-dnsop-svcb-https-03.

[305] Jörg Schwenk, Marcus Niemietz, and Christian Mainka. "Same-Origin Policy: Evaluation in Modern Browsers". In: *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, Aug. 2017, pp. 713–727. ISBN: 978-1-931971-40-9.
ISBN: 978-1-931971-40-9.

[306] Chromium Security. *Chrome Root Program Policy, Version 1.5*. https://www.chromium.org/Home/chromium-security/root-ca-policy/. (Accessed on 05/07/2024). Jan. 2024.

[307] Chromium Security. *Moving Forward, Together – Root Program Policy – Chromium Security*. https://www.chromium.org/Home/chromium-security/root-ca-policy/moving-forward-together/. Mar. 2023.

[308] Daniel Senie and Paul Ferguson. *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing*. RFC 2827. May 2000.
DOI: 10.17487/RFC2827.

[309] Spencer Sevilla, Matthew Johnson, Pat Kosakanchit, Jenny Liang, and Kurtis Heimerl. "Experiences: Design, implementation, and deployment of CoLTE, a community LTE solution". In: *The 25th Annual International Conference on Mobile Computing and Networking*. 2019, pp. 1–16.

[310] Sudheesh Singanamalla, Suphanat Chunhapanya, Jonathan Hoyland, Marek Vavruša, Tanya Verma, Peter Wu, Marwan Fayed, Kurtis Heimerl, Nick Sullivan, and Christopher Wood. "Oblivious DNS over HTTPS (ODoH): A Practical Privacy Enhancement to DNS". In: *Proceedings on Privacy Enhancing Technologies* 4 (2021), pp. 575–592.

[311] Sudheesh Singanamalla, Esther Han Beol Jang, Richard Anderson, Tadayoshi Kohno, and Kurtis Heimerl. "Accept the risk and continue: measuring the long tail of government https adoption". In: *Proceedings of the ACM Internet Measurement Conference*. 2020, pp. 577–597.

[312] Sudheesh Singanamalla, Apurv Mehra, Nishanth Chandran, Himanshi Lohchab, Seshanuradha Chava, Asit Kadayan, Sunil Bajpai, Kurtis Heimerl, Richard Anderson, and Satya Lokam. "Telechain: Bridging Telecom Policy and Blockchain Practice". In: *ACM SIGCAS/SIGCHI Conference on Computing and Sustainable Societies (COMPASS)*. 2022, pp. 280–299.

[313] Sudheesh Singanamalla, Muhammad Talha Paracha, Suleman Ahmad, Jonathan Hoyland, Luke Valenta, Yevgen Safronov, Peter Wu, Andrew Galloni, Kurtis Heimerl, Nick Sullivan, et al. "Respect the ORIGIN! a best-case evaluation of connection coalescing in the wild". In: *Proceedings of the 22nd ACM Internet Measurement Conference*. 2022, pp. 664–678.

[314] Sudheesh Singanamalla, Muhammad Talha Paracha, Suleman Ahmad, Jonathan Hoyland, Luke Valenta, Yevgen Safronov, Peter Wu, Andrew Galloni, Kurtis Heimerl, Nick Sullivan, Christopher A. Wood, and Marwan Fayed. "Respect the ORIGIN! A Best-Case Evaluation of Connection Coalescing in the Wild". In: *Proceedings of the 22nd ACM Internet Measurement Conference*. IMC '22. Nice, France: Association for Computing Machinery, 2022, pp. 664–678.
DOI: 10.1145/3517745.3561453.

[315] Marcin Skwarek, Maciej Korczynski, Wojciech Mazurczyk, and Andrzej Duda. "Characterizing Vulnerability of DNS AXFR Transfers with Global-Scale Scanning". In: *2019 IEEE Security and Privacy Workshops (SPW)*. San Francisco, CA, USA: IEEE, 2019, pp. 193–198.
DOI: 10.1109/SPW.2019.00044.

[316] Christopher Soghoian and Sid Stamm. "Certified lies: Detecting and defeating government interception attacks against SSL (short paper)". In: *Financial Cryptography and Data Security: 15th International Conference, FC 2011, Gros Islet, St. Lucia, February 28-March 4, 2011, Revised Selected Papers 15*. Springer. 2012, pp. 250–259.

[317] Joel Sommers. "A library for fast IP address lookup in Python". In: *jsommers/pytricia on GitHub* (2019).

[318] Chromium – Google Open Source. *EV UI Moving to Page Info*. https://chromium.googlesource.com/chromium/src/+/HEAD/docs/security/ev-to-page-info.md. (Accessed on 03/04/2020). Sept. 2019.

[319] Sebastian Speicher, Sasha Sirotkin, Sudeep Palat, and Alexei Davydov. "5G System Overview". In: *5G Radio Access Network Architecture: The Dark Side of 5G* (2021), pp. 37–122.

[320] Kotikalapudi Sriram, Doug Montgomery, and Jeffrey Haas. *Enhanced Feasible-Path Unicast Reverse Path Forwarding*. RFC 8704. Feb. 2020.
DOI: 10.17487/RFC8704.

[321] srsLTE. *srsLTE/srsLTE*. Nov. 2019.
URL: https://github.com/srsLTE/srsLTE.

[322] Alissa Starzak and Marwan Fayed. *The unintended consequences of blocking IP addresses*. https://blog.cloudflare.com/consequences-of-ip-blocking. (Accessed on 05/18/2024). Dec. 2022.

[323] Ben Stock, Giancarlo Pellegrino, Christian Rossow, Martin Johns, and Michael Backes. "Hey, you have a problem: On the feasibility of large-scale web vulnerability notification". In: *25th {USENIX} Security Symposium ({USENIX} Security 16)*. 2016, pp. 1015–1032.

[324] Srikanth Sundaresan, Nazanin Magharei, Nick Feamster, Renata Teixeira, and Sam Crawford. "Web performance bottlenecks in broadband access networks". In: *Proceedings of the ACM SIGMETRICS/international conference on Measurement and modeling of computer systems*. 2013, pp. 383–384.

[325] Sonesh Surana, Rabin K Patra, Sergiu Nedevschi, Manuel Ramos, Lakshminarayanan Subramanian, Yahel Ben-David, and Eric A Brewer. "Beyond Pilots: Keeping Rural Wireless Networks Alive." In: *NSDI*. Vol. 8. 2008, pp. 119–132.

[326] Chee Wee Tan, Izak Benbasat, and Ronald T Cenfetelli. "Building citizen trust towards e-government services: do high quality websites matter?" In: *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*. IEEE. 2008, pp. 217–217.

[327] Akamai Technologies. *Akamai Online Retail Performance Report: Milliseconds Are Critical. Web Performance Analytics Show Even 100-Millisecond Delays Can Impact Customer Engagement and Online Revenue*. https://blog.cloudflare.com/ecdsa-the-digital-signature-algorithm-of-a-better-internet/. Apr. 2017.

[328] Web Technology Surveys (W3 Techs). *Usage of CDNJS broken down by ranking*. 2022. URL: https://w3techs.com/technologies/breakdown/cd-cdnjs/ranking.

[329] Wayne Thayer. *Protecting our Users in Kazakhstan - Mozilla Security Blog*. https://blog.mozilla.org/security/2019/08/21/protecting-our-users-in-kazakhstan/. Aug. 2019.

[330] Martin Thomson and Cory Benfield. *HTTP/2*. RFC 9113. June 2022. DOI: 10.17487/RFC9113.

[331] Martin Thomson, Elio Damaggio, and Brian Raymor. *Generic Event Delivery Using HTTP Push*. RFC 8030. Dec. 2016. DOI: 10.17487/RFC8030.

[332] TracBot. *Tor blocked in UAE (#25137) | Issues | Legacy / Trac | GitLab*. https://gitlab.torproject.org/legacy/trac/-/issues/25137. (Accessed on 09/15/2020). Feb. 2018.

[333] Chromium Project Bug Tracker. *Issue 116982: Chromium opens useless TCP connections*. https://bugs.chromium.org/p/chromium/issues/detail?id=116982. Mar. 2012.

[334] Hannibal Travis. *Cyberspace Law: Censorship and Regulation of the Internet*. Routledge, 2013.

[335] Elisa Tsai, Deepak Kumar, Ram Sundara Raman, Gavin Li, Yael Eiger, and Roya Ensafi. "Certainty: Detecting dns manipulation at scale using tls certificates". In: *arXiv preprint arXiv:2305.08189* (2023).

[336] Liam Tung. *2G's security weaknesses are still a problem, even for modern phones*. https://www.zdnet.com/article/2gs-security-weaknesses-are-still-a-problem-even-for-modern-phones/. Jan. 2022.

[337] Sean Turner. "Transport layer security". In: *IEEE Internet Computing* 18.6 (2014), pp. 60–63.

[338] European Union. *What is considered personal data under EU GDPR*. https://gdpr.eu/eu-gdpr-personal-data/. (Accessed on 02/27/2021).

[339] Tommaso M Valletti. "Is mobile telephony a natural oligopoly?" In: *Review of Industrial Organization* 22 (2003), pp. 47–65.

[340] W3C. *Same Origin Policy*.
URL: https://www.w3.org/Security/wiki/Same_Origin_Policy.

[341] WHATWG W3C. *Cross-Origin Resource Sharing*. June 2020.
URL: https://www.w3.org/TR/2020/SPSD-cors-20200602/.

[342] Matthäus Wander. "Measurement survey of server-side DNSSEC adoption". In: *2017 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE. Dublin, Ireland: IEEE, 2017, pp. 1–9.

[343] Xiao Sophia Wang, Aruna Balasubramanian, Arvind Krishnamurthy, and David Wetherall. "Demystifying Page Load Performance with WProf". In: *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*. Lombard, IL: USENIX Association, Apr. 2013, pp. 473–485. ISBN: 978-1-931971-00-3.
ISBN: 978-1-931971-00-3.

[344] Xiao Sophia Wang, Aruna Balasubramanian, Arvind Krishnamurthy, and David Wetherall. "How Speedy is SPDY?" In: *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*. 2014, pp. 387–399.

[345] Xiao Sophia Wang, Arvind Krishnamurthy, and David Wetherall. "Speeding up web page loads with shandian". In: *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*. 2016, pp. 109–122.

[346] Zhiheng Wang. *Navigation Timing - World Wide Web Consortium (W3C)*. https://www.w3.org/TR/navigation-timing/. (Accessed on 09/17/2020). Dec. 2012.

[347] Murdoch Watney. "Law enforcement access to end-to-end encrypted social media communications". In: *7th European Conference on Social Media ECSM 2020*. 2020, p. 322.

[348] Nicholas Weaver, Christian Kreibich, and Vern Paxson. "Redirecting DNS for Ads and Profit." In: *FOCI* 2 (2011), pp. 2–3.

[349] Jason Weil, Victor Kuarsingh, Chris Donley, Christopher Liljenstolpe, and Marla Azinger. *RFC 6598: IANA-Reserved IPv4 Prefix for Shared Address Space*. 2012.

[350] Miaowen Wen, Qiang Li, Kyeong Jin Kim, David López-Pérez, Octavia A Dobre, H Vincent Poor, Petar Popovski, and Theodoros A Tsiftsis. "Private 5G networks: concepts, architectures, and research landscape". In: *IEEE Journal of Selected Topics in Signal Processing* 16.1 (2021), pp. 7–25.

[351] Andrew Whalley. *Google Online Security Blog: Protecting Chrome users in Kazakhstan*. https://security.googleblog.com/2019/08/protecting-chrome-users-in-kazakhstan.html. Aug. 2019.

[352] Gavin Wood et al. “Ethereum: A secure decentralised generalised transaction ledger”. In: *Ethereum project yellow paper* 151.2014 (2014), pp. 1–32.

[353] Diwen Xue, Reethika Ramesh, Arham Jain, Michalis Kallitsis, J Alex Halderman, Jedidiah R Crandall, and Roya Ensafi. “{OpenVPN} is open to {VPN} fingerprinting”. In: *31st USENIX Security Symposium (USENIX Security 22)*. 2022, pp. 483–500.

[354] Xynou, Maria, and Filasto, Arturò. *Iran Protests: OONI data confirms censorship events (Part 1) | OONI*. https://ooni.org/post/2018-iran-protests/. (Accessed on 09/15/2020).

[355] Kinuko Yasuda. *Signed HTTP Exchanges*. https://developers.google.com/web/updates/2018/11/signed-exchanges. Nov. 2018.

[356] Shunliang Zhang. “An overview of network slicing for 5G”. In: *IEEE Wireless Communications* 26.3 (2019), pp. 111–117.

[357] Fangming Zhao, Yoshiaki Hori, and Kouichi Sakurai. “Analysis of privacy disclosure in DNS query”. In: *2007 International Conference on Multimedia and Ubiquitous Engineering (MUE’07)*. IEEE. 2007, pp. 952–957.

[358] Fangming Zhao, Yoshiaki Hori, and Kouichi Sakurai. “Two-servers PIR based DNS query scheme with privacy-preserving”. In: *The 2007 International Conference on Intelligent Pervasive Computing (IPC 2007)*. IEEE. 2007, pp. 299–302.

[359] Liang Zhu, Zi Hu, John Heidemann, Duane Wessels, Allison Mankin, and Nikita Somaiya. “T-DNS: Connection-oriented DNS to improve privacy and security”. In: *ACM SIGCOMM Computer Communication Review* 44.4 (2014), pp. 379–380.

Vita

2012-2016: Bachelor of Technology (B.Tech), *National Institute of Technology - Warangal (NIT-W)*
2014-2016: Certificate, *Indian School of Business – Hyderabad*
2019-2024: Master of Science (M.S), *University of Washington, Seattle*
Doctor of Philosophy (Ph.D), *University of Washington, Seattle*