Towards Secure Name Resolution on the Internet (v1.1)

Christian Grothoff, Inria
Matthias Wachs, TU Munich
Monika Ermert, Heise Verlag
Jacob Appelbaum, TU Eindhoven

Abstract

The Domain Name System (DNS) provides crucial name resolution functions for most Internet services. As a result, DNS traffic provides an important attack vector for spy agencies, as demonstrated by the QUANTUMDNS and MORECOWBELL programs of the NSA. This article reviews how DNS works, and explains alternative methods designed to improve the security and privacy of domain name lookups for the future Internet.

Errata: v1.1: Corrected characterization of NSEC5.

1 Introduction

On the net, close to everything starts with a request to the Domain Name System (DNS), a core Internet protocol to allow users to access Internet services by names, such as `www.example.com`, instead of using numeric IP addresses, like 192.0.2.137 or even worse 2001:db8:4145::4242. Developed in the "Internet good old times" where privacy and security was not a concern, the contemporary DNS allows DNS operators to monitor user behavior and usage patterns, and exposes information about the existence and availability of most services on the Internet [10]. Consequently, it now attracts not only all sorts of commercially-motivated surveillance and manipulation,[1] but–as new documents of the NSA spy program MORECOWBELL confirm–also the National Security Agency as well as other intelligence agencies.

DNS currently treats all information in the DNS database as public data. The content of queries and answers is typically not encrypted. This allows passive attackers to monitor the queries of users and see which services they are using and which websites they are visiting. For an active attacker, DNS facilitates locating potentially vulnerable services, which is the first step to their subsequent exploitation with commercially available 0-day attacks.

Given the design weaknesses of DNS, this begs the question if DNS can be secured and saved, or if it has to be replaced — at least for some use cases.

In the last two years, there has been a flurry of activity to address security and privacy in DNS at the Internet Engineering Task Force (IETF), the body that specifies Internet standards, including the DNS. The Internet Architecture Board, peer body of the IETF, called on the engineers to use encryption everywhere, possibly including DNS. [8]

Despite the acknowledgment of the the DNS weaknesses and privacy implications in RFC 7626 [7] experts are not expecting that existing industry solutions will change the situation anytime soon:

> "It seems today that the possibility of massive encryption of DNS traffic is very remote." [9]

The discussions in the IETF now include proposals for "query name minimization", Confidential DNS, DNS over TLS, DNSCurve and more radical proposals for alternative name system designs to improve privacy. Additional work on encrypting traffic to the authoritative name servers high in the chain is ongoing. [12] All of these designs take different approaches in reducing the role of DNS as the ultimate source of meta data in the digital panopticon known as the Internet. Before we present the different approaches, we illustrate the security goals and the threat model, using the NSA spy programs as a highly capable attacker and explain what the benefits for the attacker and the risks for the DNS user are. Note that, the NSA is only

[1] For example, Google's public DNS service permanently logs a dozen items about each request, including the requested domain name, see `https://developers.google.com/speed/public-dns/privacy`. Also, Cisco-owned OpenDNS logs " *any statistical information related to the usage, traffic patterns and behavior of the users*", see `https://www.opendns.com/terms-of-service/`. Finally, there are ISPs manipulating DNS requests and responses, thereby achieving monetary benefits through advertisements, see `https://www.wired.com/2008/04/isps-error-page`. Security problems of these "wildcard" redirections of DNS traffic have been noted, but are ongoing. [1]

one of the potential attackers, as other state actors as well as criminals can use the same techniques, and some commercial entities mine data as well to feed their profiling databases. We present the NSA attack as an exemplary, because of their technical capabilities and the explanations of their DNS attack strategies published in recently published documents of the agency itself.

2 Background: DNS

The Domain Name System (DNS) is an essential part of the Internet as it provides mappings from host names to IP addresses, providing memorable names for users. DNS is hierarchical and stores name-value mappings in so-called *records* in a distributed database. A record consists of a name, type, value and expiration time. Names consist of *labels* delimited by dots. The root of the hierarchy is the empty label, and the right-most label in a name is known as the top-level domain (TLD). Names with a common suffix are said to be in the same *domain*. The *record type* specifies what kind of value is associated with a name, and a name can have many records with various types. The most well-known record type is the "A" record, which maps names to IPv4 addresses.

The DNS database is partitioned into *zones*. A *zone* is a portion of the namespace where the administrative responsibility belongs to one particular authority. A zone has unrestricted autonomy to manage the records in one or more domains. Very importantly, an authority can delegate responsibility for particular *subdomains* to other authorities. This is achieved with an "NS" record, whose value is the name of a DNS server of the authority for the subdomain. The *root zone* is the zone corresponding to the empty label.

It is managed by the Internet Assigned Numbers Authority (IANA), which is currently operated by the Internet Corporation for Assigned Names and Numbers (ICANN), which was depoliticized in 2016 and is since no longer under the control of the National Telecommunications and Information Administration (NTIA) but instead subject to a complex global multistakeholder oversight process where ordinary users will have a hard time being involved.[17]

The root zone contains "NS" records which specify names for the authoritative DNS servers for all TLDs, such as ".de" or ".berlin".

Names in DNS are resolved using *resolvers*. Many modern operating systems do not provide a full implementation of a DNS resolver but only so called *stub resolvers*. These stub resolvers do not resolve names directly but forward the request to another resolver. In general, we will refer to resolvers that merely forward requests (and possibly cache replies) as *forward resolvers*. After forwarding, the query eventually reaches a *recursive name server*, which is typically provided by the Internet Service Provider (ISP), as shown in Figure 1. These recursive name servers resolve the name by first querying the root servers for the required name and by way of recursion go down the DNS tree to fetch the information from the authoritative DNS server. The queried root servers provide the querying resolver with an "NS" record to the server authoritative for the TLD zone, the authoritative server for the zone provides the record for the authoritative server for the domain, subdomain and so on. This *iterative* process is repeated, and terminates for sure when the resolver queries the *authoritative name server* which is responsible for a particular domain.

DNS strongly benefits from caching of DNS information: many *caching resolvers* store information previously requested to improve lookup performance. They use cached record data to skip some or all of the iterations, and thus can return information more quickly to the client.

With the use of forwarding resolvers, the IP address of the client is hidden from authoritative name servers. This gives the user a certain degree of privacy as it prevents operators of authoritative name servers to monitor the source of DNS requests. Naturally, the operators of the forwarding resolvers can still trivially monitor and censor users' requests. Passive dragnet monitoring with systems such as TURMOIL and XKEYSCORE are also able to see any part of the transaction that is available in the ingestion filter.

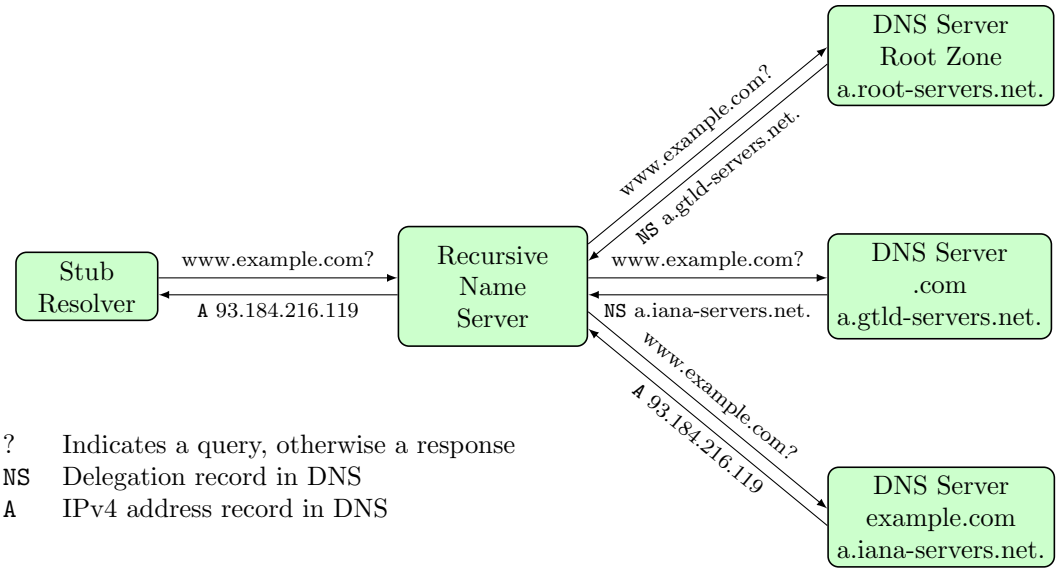


Figure 1: Resolving the name `www.example.com` with DNS. Many operating systems only provide minimal *stub resolvers* forwarding requests to full resolvers. To resolve a name, these resolvers start with querying the name servers of the root zone. If a server cannot provide the required information, it refers the resolver to the next server to query until the server *authoritative* for the respective zone is found.

3 Security goals

When considering improving the security of DNS, there have been striking disagreements among designers as to what the security goals of the DNS system should be. What most designers do agree with is that for the public DNS service, anyone should be able to resolve domain names in it without prior authorization. This does not preclude the possibility of DNS servers returning sensitive records only for certain users, an approach commonly known as *split view*. However, generally speaking, the consensus is that DNS should answer queries without requiring origin authentication.

3.1 Query origin anonymity

However, even if users of DNS do not have to authenticate, that does not mean that they are anonymous. In the original protocol, the IP addresses of the stub resolvers are hidden behind the recursive name servers, providing a thin veil of privacy. However, this may come at the expense of the origin having to trust the recursive name solver. Furthermore, with the introduction of the client subnet extension [13], recursive name servers may be configured to expose most of a client's IP address to other DNS servers.

3.2 Data origin authentication and integrity protection

Except for regional censors that today block domains by modifying DNS responses, most designers want to see the authenticity and integrity of DNS responses protected. Weak designs simply use secure communication channels between authenticated resolvers. This achieves integrity protection against adversaries in the network but does not help with data authenticity. Another possibility is to cryptographically sign responses with private keys held online; however, as a strong adversary may compromise authoritative name servers, the best protections are achieved by using offline keys for signing zone data to achieve "end-to-end" security including origin authenticity and integrity protection.

3.3 Zone confidentiality

Before the DNS, all name resolution data was public. With DNS, the notion that zone data could be semi-private and only be exposed upon matching request became a possibility. Exposing full zone information provides useful information to attackers, as they can enumerate network services offered by the target, which with virtual hosting or IPv6 might otherwise not be feasible. Thus, it is desirable to minimize the adversary's ability to enumerate the names in a zone.

3.4 Query and response privacy

Finally, the DNS query itself or the DNS response may include sensitive information. The design principle of data minimization dictates that participants should only learn as much as necessary, thus some proposals try to make DNS less chatty. In the most extreme case, a domain name may contain a password, and responses might contain key material, which both ought to be kept confidential from the recursive and (online) authoritative name servers.

3.5 Censorship resistance

A special goal of some name systems is resistance against censorship. The goal is to make it impossible even for governments that have jurisdiction over any possible DNS operator to block name resolution using legal attacks. This is typically achieved by designs that are self-organizing and thus do not require the interaction with professional registrars.

4 Exemplary Attacker: The NSA's MORECOWBELL and QUANTUMDNS programs

These security goals are critical, as the respective threats against the DNS and its users are not theoretical. As set of top secret documents published by Le Monde [18] revealed, the American spy agency NSA monitors DNS as a source of information about the Internet (Figure 3). NSA's MORECOWBELL program uses a dedicated covert monitoring infrastructure to actively query DNS servers and perform HTTP requests to obtain meta information about services and to check their availability (Figure 2).

Despite the open nature of DNS, the NSA does so covertly (Figure 4) to ensure the thousands of DNS lookups every hour are not attributed to the US government (USG). In fact, the servers the NSA rented for the purpose of monitoring DNS and checking Web servers using HTTP are located in Malaysia, Germany and Denmark (Figure 5), allowing the NSA to perform the monitoring covertly and to get a more global view on DNS name resolution and service availability. While the NSA slides only list these three countries, the PACKAGEDGOODS non-attributable monitoring infrastructure that MORECOWBELL builds on is known to span machines in at least 13 other countries, as described previously by Der Spiegel in a set of slides describing the NSA's TREASUREMAP program. [29]

What is interesting is that at the time, the NSA did not care much about the specific content of the Web servers or the DNS entries — as usual the NSA is after the meta data: The NSA wants to know if the DNS information has changed, and check on the availability of the service. The slides show that this simple check has some rather benign uses, for example it is used to monitor some of the US government's own websites.

A key justification for the need to make the active probing of DNS unattributable to the US government is most likely its use for "Battle Damage Indication" (Figure 6). Specifically, after "Computer Network Attacks (CNA)" are used against critical network infrastructure, the US may use such probes to confirm that its attacks have found their targets when the lights go out on the Internet systems, say of some foreign government. By monitoring for changes in the DNS, the attack could be repeated if the victim tries to shift its services to another system or network. By keeping the monitoring infrastructure covert and geographically distributed, the NSA gets a global view on the impact of an attack. This makes it harder for victims to

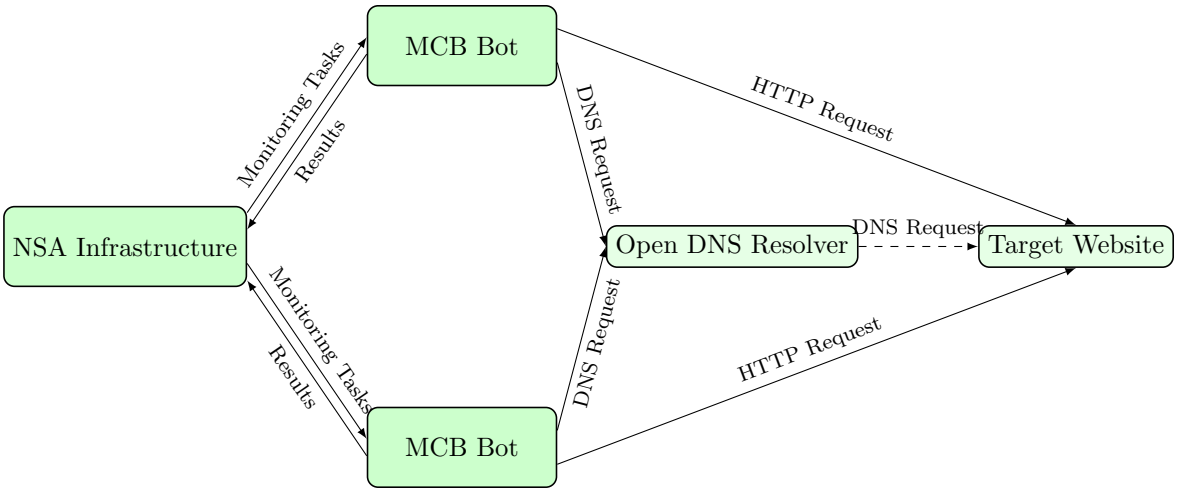


Figure 2: From [18]: NSA's MORECOWBELL infrastructure: a list of targets to monitor is deployed to geographically distributed bots performing DNS and HTTP requests against target websites to collect information about the availability of services. The resulting data are returned to the NSA in regular intervals.

identify the monitoring servers, which otherwise might enable victims to evade the attack by treating requests from monitors differently.

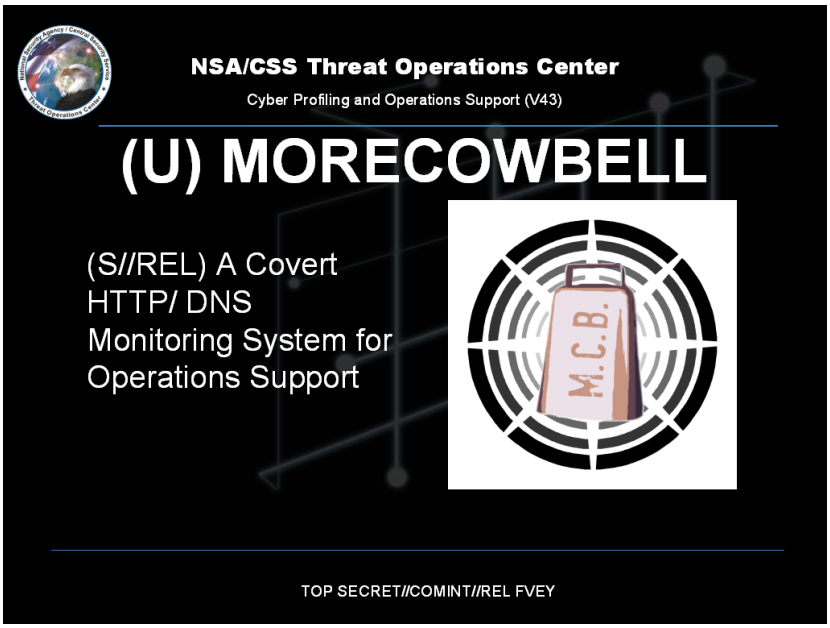


Figure 3: From [18]: MORECOWBELL: A Covert HTTP/DNS Monitoring System

The various documents of the NSA relating to DNS show that existing covert attacks on DNS enable mass surveillance and active attacks. [38] With the revelation about the NSA's QUANTUMTHEORY family of projects (Figure 7) with subprojects like QUANTUMDNS (Figure 8), we know that powerful attackers like nation states can not only eavesdrop DNS traffic but also inject DNS responses to modify the result of name resolution or make it even completely fail. [32] With DNS not providing confidentiality to protect a user's

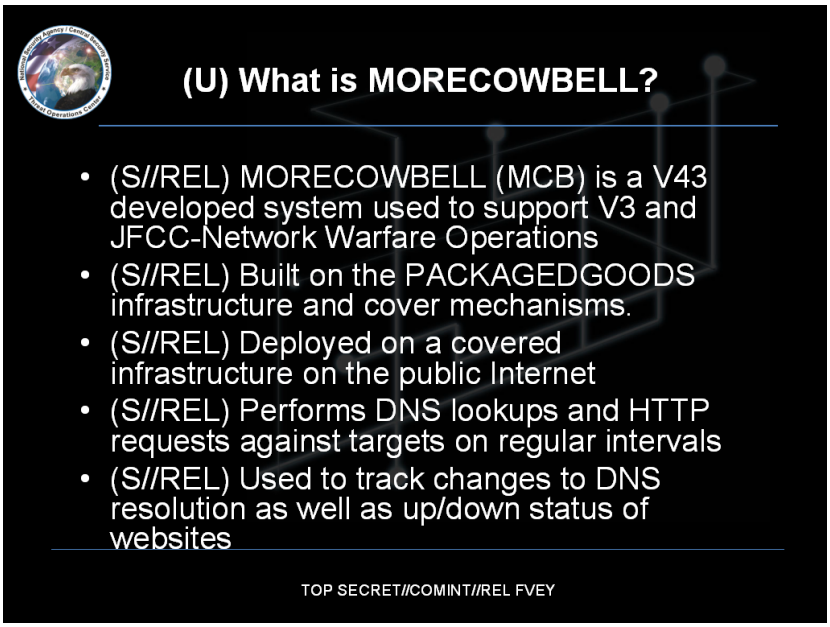


Figure 4: From [18]: What is MORECOWBELL.

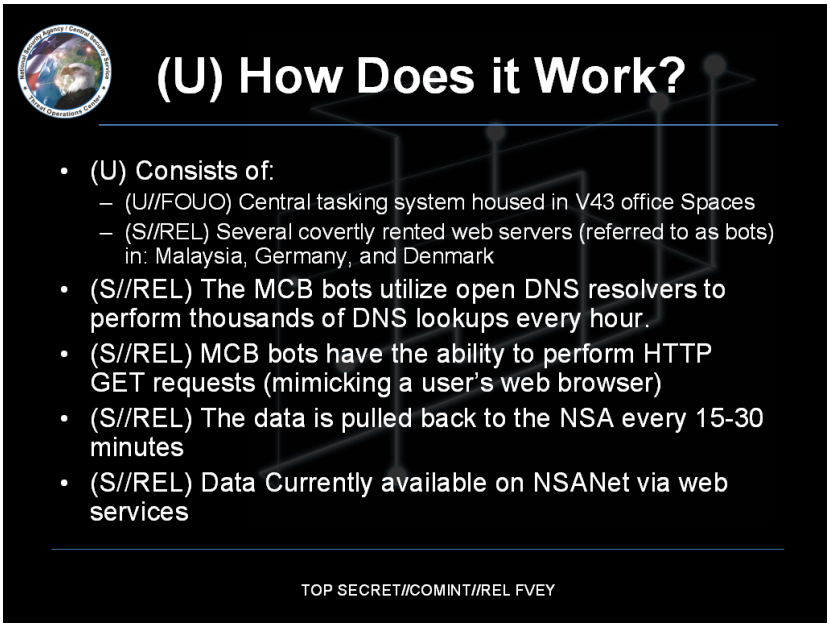


Figure 5: From [18]: How does MORECOWBELL work?

privacy, it is easy to create a profile of the users and their surfing behavior on the Web. [24] This information could then also be used to perform QUANTUMTHEORY attacks against the target. NSA programs like QUANTUMBOT have the purpose to monitor IRC botnets and detect computers operating as bots for a botnet and hijack the command and control channel to manage the bots. These programs are evaluated by

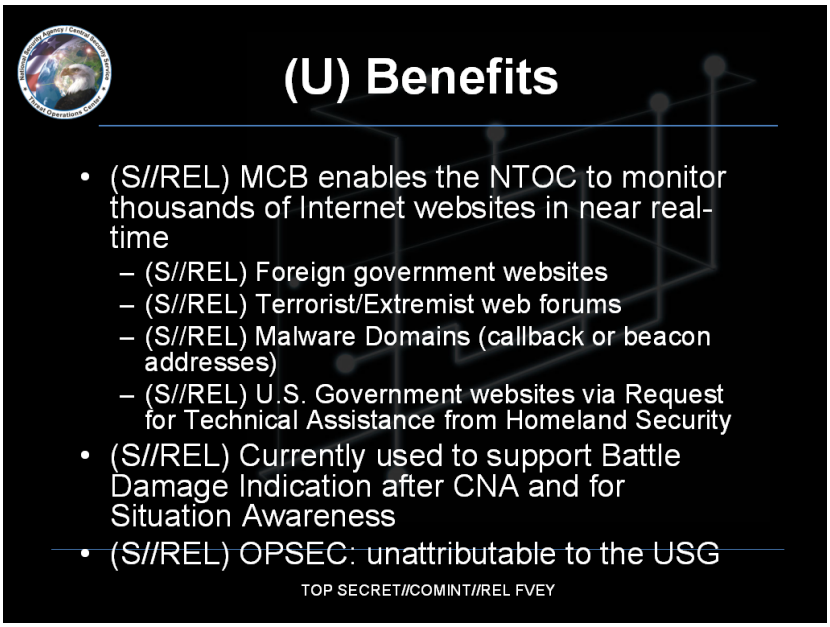


Figure 6: From [18]: "Benefits" of MORECOWBELL.

the NSA to be *highly successful* according to their documents. [28]

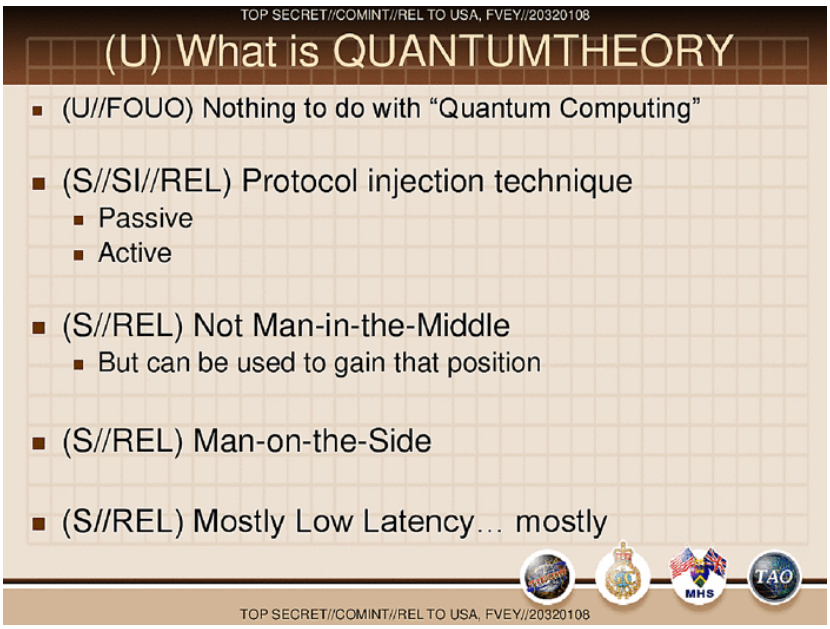


Figure 7: NSA's QUANTUMTHEORY: The man-on-the-side attack.

Thus, the Internet community needs to work towards resolving the privacy and security issues with name resolution and the current Domain Name System (DNS). In the next step, we will review a range of current proposals that have been made to improve the security of this critical Internet service.

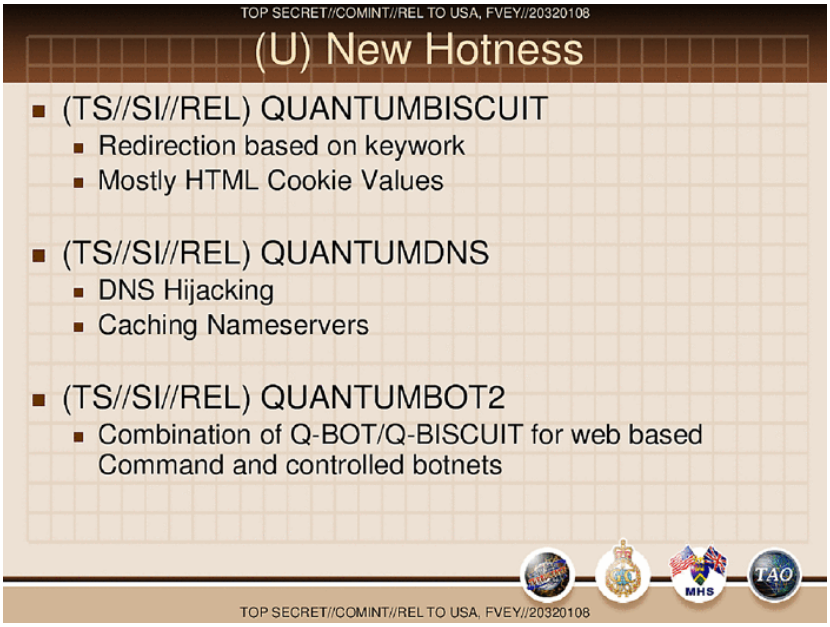


Figure 8: NSA's QUANTUMDNS: Attacks on DNS are not theoretical. Other slides from the NSA say that QUANTUMDNS is operational and has been successfully used.

5 Adversary Model

To evaluate existing approaches aiming to improve name resolution security and privacy, we employ two different adversaries:

On the one hand, we examine adversaries within the name system. This can be DNS infrastructure providers operating DNS relevant systems including DNS recursive or forward resolvers. Such adversaries can be honest-but-curious interested in users' usage patterns by monitoring name resolution. To counteract such an adversary query origin anonymity and query response privacy are relevant security goals. Besides being curious, such an adversary may be interested in modifying results or make name resolution fail, requiring integrity protection, data origin authentication, and censorship resistance as security goals to antagonize such an attacker.

On the other hand, we employ very powerful adversaries as introduced with the NSA and its MORECOWBELL and QUANTUMDNS programs. Such adversaries may be interested in monitoring users' behavior monitoring DNS resolution by being able to eavesdrop network traffic, requiring query origin anonymity and query response privacy as a countermeasure. Besides monitoring, such adversaries may want to tamper with name resolution by modifying name resolution (requiring integrity protection and data origin authentication as security goals) or make name resolution fail using technical or legal means (requiring censorship resistance for name systems). Such adversaries may exploit name systems by obtaining zone information to learn about network services that they may subsequently target and exploit. Here, zone confidentiality and response confidentiality are important to avoid leaking knowledge about potential targets.

6 DNSSEC

The Domain Name System Security Extensions (DNSSEC) [2] add integrity protection and data origin authentication for DNS records. DNSSEC does not attempt to improve privacy. It adds record types for public keys ("DNSKEY"), signer delegation ("DS"), for signatures on resource records ("RRSIG") and secure denial of existence ("NSEC"). Figure 9 illustrates the interactions among resolvers using DNSSEC.

DNSSEC creates a hierarchical public-key infrastructure in which all DNSSEC operators must participate. It establishes a trust chain from a zone's authoritative server to the trust anchor, which is associated with the root zone. This association is achieved by distributing the root zone's public key out-of-band with, for example, operating systems. The trust chains established by DNSSEC mirror the zone delegations of DNS. With TLD operators typically subjected to the same jurisdiction as the domain operators in their zone, with respect to censorship resistance these trust chains are at risk of attacks using both legal and technical means.

Current DNSSEC deployment suffers from the use of the RSA crypto system, which thus must be supported by every DNSSEC-enabled resolver. The use of RSA leads to unnecessarily large keys and signatures, and the effect is amplified because response includes the signatures for all of the signature schemes supported by the authoritative server. This can result in message sizes that exceed traditional size restrictions on DNS packets, leading to additional vulnerabilities [20]. While the IETF has started to add additional ciphers based on elliptic curves [21], deploying multiple ciphers further increases packet size and computational cost (if both ciphers are used to secure the same delegation), or reduces security to the weaker of the two ciphers if a mixture of ciphers is used on the resolution path.

DNSSEC also effectively lifts the few traditional limitations on bulk acquisition of zone data, practically reducing zone confidentiality. Before DNSSEC, DNS zone administrators could disallow zone transfers, making it difficult for an adversary to systematically enumerate all of the DNS records in a zone. However, as DNS allows for negative replies (NXDOMAIN), DNSSEC needed a way to create a signed statement that records did not exist. As DNSSEC was designed to keep the signing key offline, "NSEC" records were introduced to certify that an entire range of names was not in use. By looking at the boundaries of those ranges, an adversary can quickly enumerate all names in a zone that are in use. An attempt to fix this via the introduction of "NSEC3" records has been described as broken by security researchers[2]. Nevertheless, NSEC3 is now widely used.[3] As a result, DNSSEC makes it even easier for an adversary to discover vulnerable services and systems. [4] But above all, zone confidentiality remains a desideratum.

In the following section we describe the different approaches to add confidentiality to the DNS.

7 Query name minimization

The recent discussions in the IETF to improve privacy in DNS (discussed in the DNSOP and DPRIVE working groups) include a standard for so-called *query name minimization* or *QNAME minimization* [11], which is easy to implement as it does not actually require changes to the DNS protocol. Query name minimization would slightly improve query privacy by having recursive name servers not send the full query to the DNS servers contacted in each resolution step. Instead, each DNS server only receives as much of the DNS name as is necessary for making progress in the resolution process (Figure 10). Consequently, the full name being queried is typically only exposed to the final authoritative DNS server.

Query name minimization can simply be implemented by changing how recursive name servers construct their iterative queries. Query name minimization may negatively impact performance, as at least in theory the full query may enable the DNS servers to respond faster with the ultimate answer, if cached information is available or they are the authoritative server for the queried fully qualified domain name. Even with query name minimization, the recursive name servers (at an ISP for example) still learn the full query and reply of a user.

Query name minimization has the advantage that its deployment only requires changes to the recursive name server, and the disadvantage that the change is entirely outside of user control. Query name minimization can be combined with the various approaches to encrypt DNS traffic presented in the next sections. Without query name minimization, simply encrypting DNS traffic— for example using TLS as described in the following section— continues to expose the full query to many DNS servers, in particular root servers and authoritative servers for the respective TLD. With query name minimization, it is possible that only the recursive name server and the authority for the full domain name learn the full name.

[2] `https://dnscurve.org/espionage2.html`

[3] `http://secspider.verisignlabs.com/stats.html`

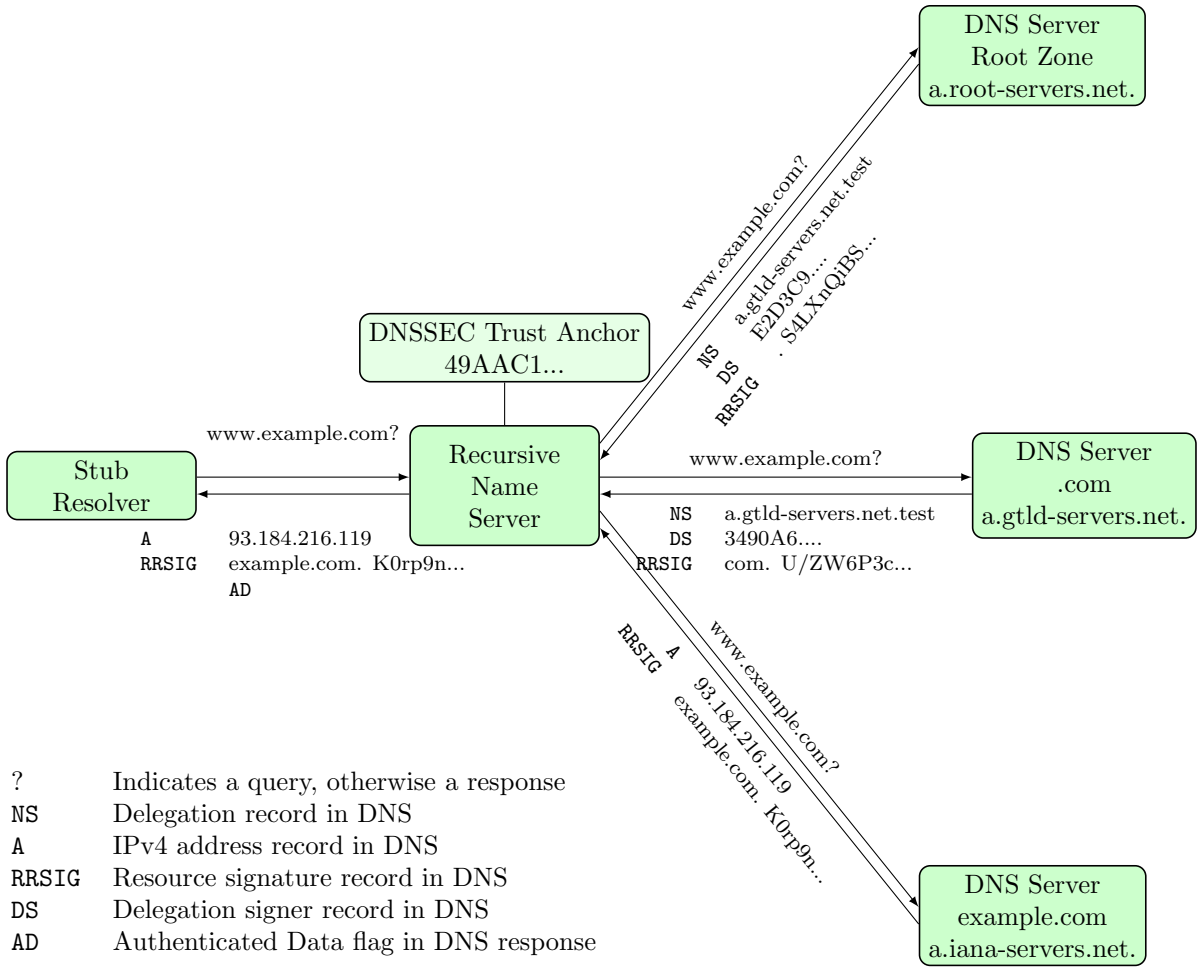


Figure 9: Resolving the name `www.example.com` with DNS and DNSSEC: information returned by name servers is cryptographically signed to ensure authenticity and integrity. This information is stored in "RRSIG" records and information about the parent zone stored in "DS" records. A resolver can verify a signature by following this trust chain and using the *trust anchor* shipped out-of-band. Stub resolvers cannot verify this chain and the resolver therefore indicates to the stub resolver that it checked authenticity by setting the `AD` bit in the response given to the client.

8 DNS-over-TLS

Discussions to use Transport Layer Security (TLS) for encrypting DNS traffic were previously often rejected because of the performance loss associated with such a change. In the discussions about DNS over TLS (standardized as RFC 7858 [23]) it was pointed out that using TLS would not only be beneficial in supporting query and response privacy and hop-by-hop integrity protection, but by switching to TCP — and therefore from connectionless UDP to connection-oriented TCP — might also help mitigate against amplification attacks on (or by) DNS servers. [25]

By re-using a TCP connection for multiple DNS requests with moderate timeouts, pipelining requests and allowing out of order processing, the DNS-over-TLS proposal promises reasonable performance despite the overheads from TCP and TLS.

However, even if TLS were to be deployed for DNS, this would not improve query origin anonymity since it still leaks meta data, allowing third parties to easily determine which DNS data a user accesses: In the

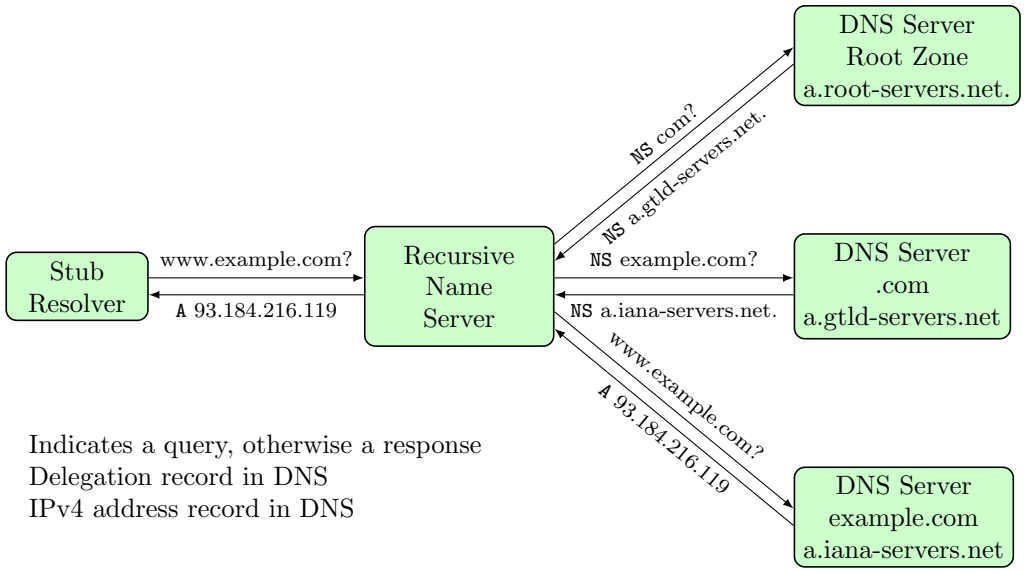


Figure 10: With query name minimization, resolving the name `www.example.com` no longer exposes the full name and query type to the root zone and the `.com` authority. Naturally, this scheme still leaks quite a bit of sensitive information to the TLD's DNS server, but less (no `www` in our example) than otherwise. Furthermore, the effect is even weaker in practice, as root zone is already often not contacted as information about TLD name servers is typically cached at forwarding resolvers.

IETF proposal, TLS is used in combination with the traditional DNS lookup paths, which may involve the use of forward resolvers that assist endpoints performing DNS queries. The involvement of such forward resolvers can obscure the user's IP address from the other DNS servers; naturally, for this to be sufficient the forward resolvers themselves would have to be trusted to not spy on the user. Furthermore, TLS itself does not have the best security track record, with dozens of issues in recent years ranging from high-profile certificate authority compromises to broken implementations and insecure cipher modes. [22] Ways for users to configure just how broken (or optimistic [16]) TLS is allowed to be for their DNS-over-TLS requests continues to be the subject of a current IETF draft [15]. Key problems in this context include the need for incremental deployment, and that TLS authentication itself can require DNS names [33] or even use DNS records [3], resulting in a bootstrap problem that needs to be mitigated.

TLS is not the only possible method for encrypting DNS queries and replies as they traverse the network to increase query and response privacy as well as integrity. DNSCurve and Confidential DNS are alternative proposals to protect the content of DNS queries and replies from network-level monitoring and modification.

DNS-over-TLS is available the Unbound DNS server[4] and the Knot resolver[5]. It is also possible to implement DNS-over-TLS using a TLS proxy in front of a nameserver. Several pilot public servers implementing DNS-over-TLS are currently set up[6] one for example at the Domain Name System Operations Analysis and Research Center.[7]

9 DNSCurve

The first practical system that improves confidentiality with respect to DNS queries and responses was DNSCurve [6]. In DNSCurve, session keys are exchanged using Curve25519 [5] and then used to provide

[4] `https://unbound.net/`, retrieved February 2017.
[5] `https://www.knot-resolver.cz/`, retrieved February 2017
[6] `https://portal.sinodun.com/wiki/display/TDNS/DNS-over-TLS+test+servers` contains a list, retrieved February 2017.
[7] `https://www.dns-oarc.net/oarc/services/dnsprivacy`, retrieved February 2017.

authentication and encryption between caches and servers. DNSCurve improves the existing Domain Name System with query and response confidentiality and hop-by-hop integrity without the need to create expensive signatures or (D)TLS sessions. Specifically, DNSCurve achieves the same round trip time (RTT) as DNS by embedding the public key of the server in the "NS" record, conflating the DNS namespace with key information.

DNSCurve creates an authenticated and encrypted association between a *DNSCurve server* and a *DNSCurve cache*, the latter being a caching recursive DNS resolver running at the endpoint instead of a DNS stub resolver (Figure 11). As DNSCurve does not use signatures, the DNSCurve cache cannot prove the authenticity of the cached records to other users, limiting the utility of each cache to the respective endpoint.

While in DNSCurve the user no longer has to trust a forward resolver, the endpoint's IP address is now directly exposed to the authoritative DNS servers: it is no longer obscured by recursive name servers operated by network service providers. Thus, DNSCurve can increase privacy against an adversary monitoring DNS traffic on intermediary systems or with other cable tapping, but reduces query origin anonymity with respect to authoritative DNS servers, as they learn both the full query and the identity (IP address) of the user. Another commonly voiced concern about DNSCurve is the need to keep private keys online. DNSCurve also cannot protect against censorship, as certain governments continue to effectively control the hierarchy of registrars and can thus make domains disappear. With respect to attacks from the NSA, DNSCurve only helps users against passive surveillance on the wire by protecting the confidentiality of at least the DNS payload.

With DNSCurve, DNS servers remain a juicy target for mass surveillance. Furthermore, as with DNS, the well-known and easily located DNS servers remain a target and confirmation vector for attacks on critical infrastructure. With DNSCurve, the need for online public key cryptography by the DNS authorities may open up an additional vulnerability to computational denial of service attacks if a small CPU is used to handle a high-speed link.

DNSCrypt

DNSCrypt is an unstandardized but documented protocol largely based on DNSCurve. It protects the end user's stub resolver queries from network surveillance and tampering hereby improving query and response privacy and integrity. As it is based on DNSCurve, it does not solve any of the major other privacy or security issues present in DNS. The largest known resolver to support DNSCrypt is OpenDNS. There are a number of open DNSCrypt resolvers run by the DNSCrypt community. Today, DNSCrypt remains the most widely deployed DNS encryption protocol designed to prevent surveillance of end users from the network. However, it only helps to solve half of the privacy problem, and it is not widely adopted or standardized.

10 Confidential DNS

Another IETF draft which has been discussed in the IETF DPrive Working Group suggests an alternative method for adding encryption to DNS. It uses the main extension mechanism of DNS, the introduction of additional record types, to encrypt DNS traffic [39], hereby achieving query and response privacy and integrity protection. With Confidential DNS, a new "ENCRYPT" record type is introduced to provide the necessary public key that would allow the recursive name server to encrypt the connection to the DNS server. This "ENCRYPT" record contains the public key of the DNS server to be used to encrypt communication initiated by the resolver. The "hack" used by DNSCurve where the public key was added into the "NS" response of the delegating zone is avoided.

The current draft supports two different operation modes: an *opportunistic* mode which is easier to realize since it does not require major changes to DNS infrastructure and an *authenticated* mode, where a domain's public keys are also stored in the respective parent zone, thus requiring support from the parent zone's DNS infrastructure.

With the opportunistic mode, the public key is no longer associated with the parent zone and instead served separately in the clear and possibly without authentication as a record with the target zone. As a

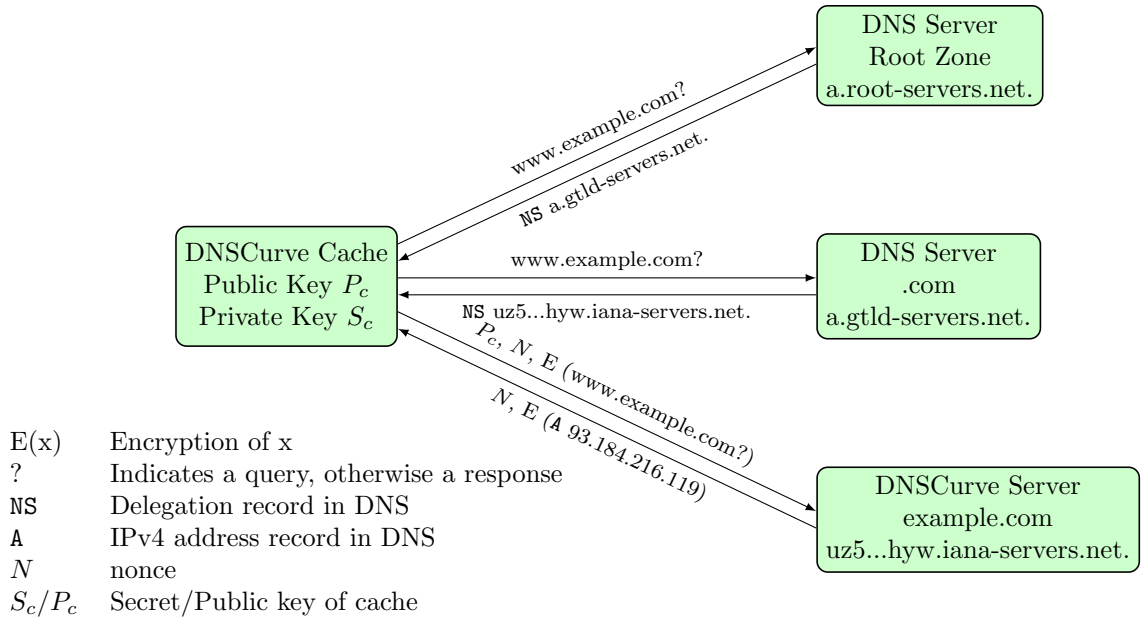


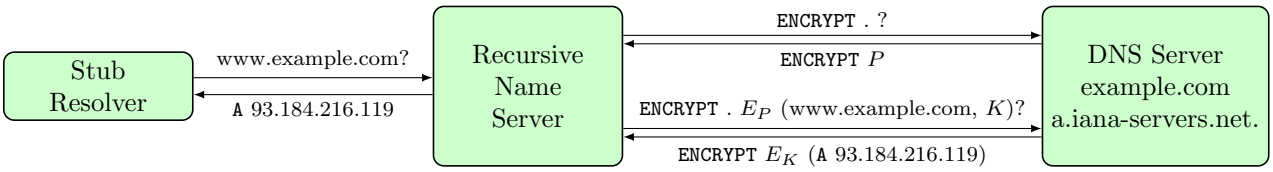
Figure 11: Resolving the name `www.example.com` with DNSCurve. With DNSCurve, the resolving cache and the DNSCurve server exchange a shared secret to encrypt their communication. The DNSCurve server's public key is encoded in the name of the name server itself using Base32. When a DNSCurve cache resolves a name and finds the name server to support DNSCurve, the cache creates a shared secret based on the server's public key, the cache's private key, and a one-time nonce. The cache sends its public key, the nonce and the query encrypted with the shared secret. The server will respond with the result of the query encrypted with the shared secret. The first two lookups to the root zone and the ".com" TLD do not use DNSCurve in the illustration as those currently do not support DNSCurve.

result, Confidential DNS using the "ENCRYPT" record only supports so-called *opportunistic encryption*, which is encryption that is trivially bypassed by a man-in-the-middle attack, as it uses unauthenticated keys for encryption.

The use of a new record type also creates the opportunity for the necessary complexity of a committee-engineered solution: Confidential DNS can use symmetric or asymmetric cryptography, and sports support for 512-bit RSA and AES in CBC mode (which was recently used to finally kill off SSL3 [26]). The draft fails to set a strong minimum baseline and to ensure that this minimum will be updated to reflect new security considerations in due course.

The draft on Confidential DNS provides also a method to achieve "real" authenticated encryption by storing a domain's public key in the respective parent zone. To do so, Confidential DNS extends DNSSEC's Delegation Signer ("DS") resource records to provide the encryption key for the zone. This resembles the "NS" record used by DNSCurve. This approaches makes Confidential DNS susceptible to censorship attacks since it relies DNS's hierarchical architecture.

The draft provides for a variety of failure modes, such as "fallback to insecure" allowing clients to relapse to insecure modes with "leaps of faith" even after secure connections used to be available. Confidential DNS allows implementations to "fallback to insecure" in case one side uses cryptographic algorithms that the other does not support. These various scenarios in which Confidential DNS simply falls back to unencrypted channels (without any indication to the user) highlight how much the design focuses on being easy to deploy at the expense of providing predictable security. Given the recent adoption of DNS-over-TLS and critiques that Confidential DNS introduces a DDoS vector, the Confidential DNS specification has not been updated in a while and remains unfinished IETF work.



?	Indicates a query, otherwise a response
.	Query for the root zone
P	Public key of server
K	Encryption Key
E_P (x)	Encryption of x with P
E_K (x)	Encryption of x with K
`A`	IPv4 address record in DNS
`ENCRYPT`	Encrypt record in DNS

Figure 12: Resolving the name `www.example.com` with opportunistic Confidential DNS. The resolver retrieves the DNS servers public key querying for the new "ENCRYPT" record. This public key can then be used to encrypt the query to the server. The resolver sends the query encrypted with the server's public key containing the query and the key to encrypt the reply with.

11 Namecoin

None of the approaches presented so far are designed to withstand legal attacks. Depending on their reach, governments, corporations and their lobbies can legally compel operators of DNS authorities to manipulate entries and certify the changes. Hence the above systems are vulnerable to censorship.

Alternative peer-to-peer name systems provide more radical solutions to secure name resolution. Timeline-based systems in the style of Bitcoin [27] have been proposed to create a global, secure and memorable name system [35]. Here, the idea is to create a single, globally accessible timeline of name registrations that is append-only. Timeline-based systems rely on a peer-to-peer network to manage updates and store the timeline. In the Namecoin system [36], modifications to key-value mappings are attached to transactions which are committed to the timeline by mining. Mining is the use of brute-force methods to find (partial) hash collisions with a state summary (fingerprint) representing the complete global state — including the full history — of the timeline.

Given two timelines with possibly conflicting mappings, the network accepts the timeline with the longest chain as valid, as it represents the largest expense of computational power. This is supposed to make it computationally infeasible for an adversary to produce an alternative valid timeline. This assumes limited computational power and may not actually be binding for certain adversaries.

To perform a lookup for a name with Namecoin, the client has to check the timeline if it contains an entry for the desired name and check the timeline for correctness to ensure that the timeline is valid. To do so, the user has to possess a full copy of the timeline (Figure 13), which had a size of about 4.7 GB in November 2016.[8] Alternatively, users may use a trusted name server participating in the Namecoin network.

Namecoin can improve user privacy if the full block chain is replicated at the user's end system. In this case, resolving a name does not involve the lookup and is thus perfectly private with respect to query origin anonymity and query and response privacy. However, replicating the full block chain at each user may be impractical for some devices should Namecoin ever grow to be a serious competitor for DNS. Namecoin also does not protect the zone information from monitoring, and in particular zone enumeration is trivial.

[8] `https://bitinfocharts.com/namecoin/`

However, the decentralized nature of Namecoin does ensure that at least battle damage indication against a name server no longer makes sense.

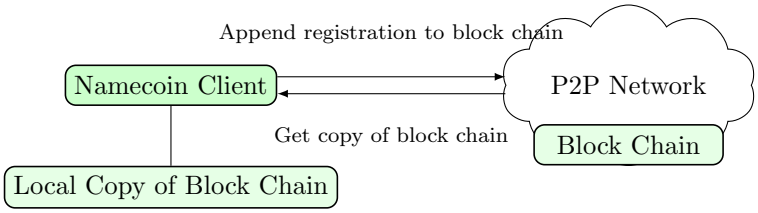


Figure 13: The Namecoin name system is decentralized and uses a peer-to-peer network. To achieve a consensus about names registered, Namecoin uses a *block chain* stored in the peer-to-peer network. To register a name, clients have to pay a miner to perform some computational work to get their name appended to the chain. To resolve a name, clients have to possess a full copy of the block chain and search for the name to resolve in the block chain.

12 The GNU name system

The authors of this article are working on the GNU Name System (GNS) [37], which is a more radical proposal to address DNS privacy and security issues, and which like Namecoin significantly departs from DNS's name resolution process. The GNS resolution process does not use resolvers querying DNS authorities. Instead, GNS uses a peer-to-peer network and a distributed hash table (DHT) to enable resolvers to lookup key-value mappings. As a result, GNS will inherit the performance and availability characteristics of the underlying DHT. Various implications of such a transition on availability and performance have been analyzed previously in [30]. However, in contrast to previous work that proposed to simply replicate information from DNS into a DHT to improve resilience and performance [31, 14], GNS provides a fully decentralized name system which is conceptually independent from DNS.

GNS is privacy-preserving since queries and responses are encrypted such that even an active and participating adversary can at best perform a confirmation attack, and otherwise only learn the expiration time of a response. Note that the queries and responses themselves are encrypted, not the connections between a resolver and some authority. As all replies are not just encrypted but also cryptographically signed, GNS provides integrity protection since peers in the DHT cannot tamper with the results without immediate detection and data origin authentication.

Due to the use of a DHT, GNS avoids DNS complications such as glue records and out-of-bailiwick lookups. In GNS, the labels of a name correspond precisely to the lookup sequence, making the complete trust path obvious to the user. Finally, the use of a DHT to distribute records also makes it possible for GNS authorities to operate zones without visible, attributable critical infrastructure that could be used for battle damage indication.

GNS can securely resolve names to any kind of cryptographic token. Thus, it can be used for addressing, identity management and as an alternative for today's battered public key infrastructures.

12.1 Names, zones and delegations

A GNS zone is a public-private key pair and a set of associated records. The GNS name resolution process basically resolves a chain of public keys. In the absence of a widely recognized and operational *root zone*, but also as an inherent alternative to hierarchical addressing, GNS uses the pseudo-TLD ".gnu" to refer to the user's own zone, which is called the *master zone*. Each user can create any number of zones, but one must be designated as the master zone. Users can freely manage mappings for the labels in their zones. Most importantly, they can delegate control over a subdomain to any other zone (including those operated by other users) using a "PKEY" record, which simply specifies the public key of the target zone. "PKEY" records are

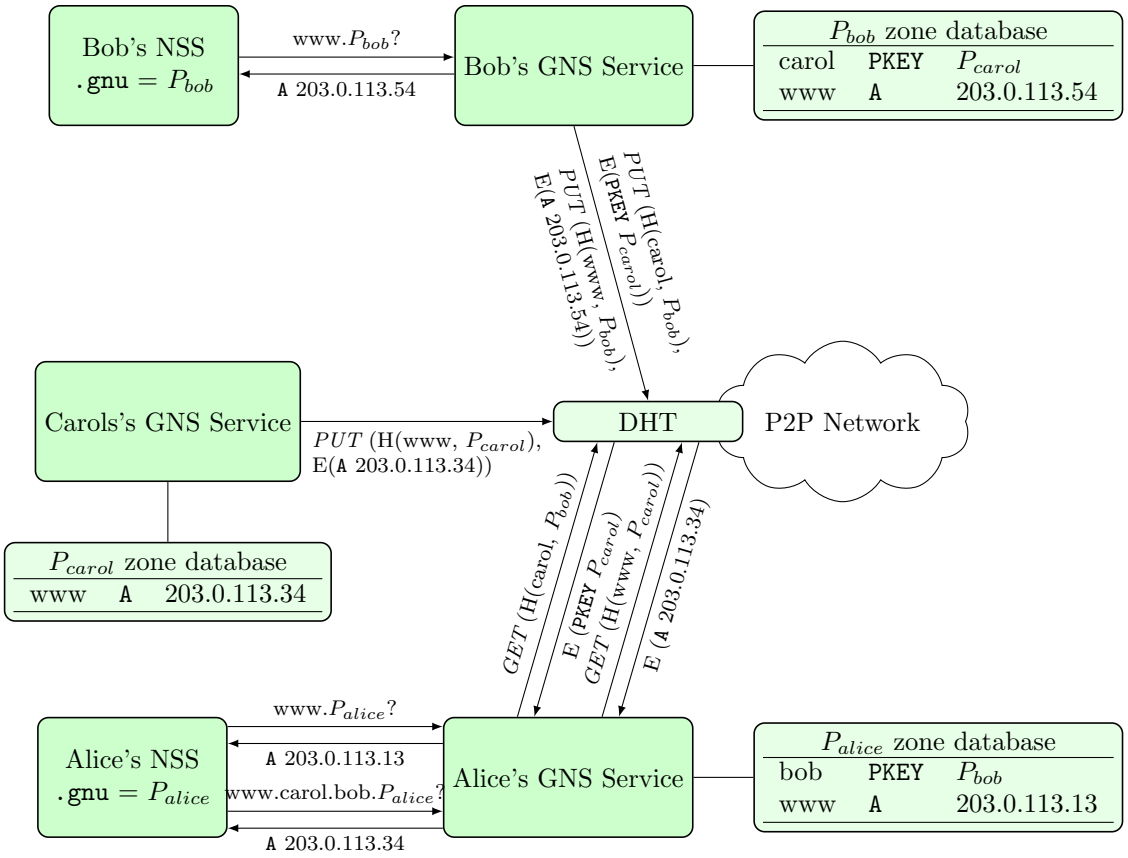


Figure 14: The GNU name system: with GNS, every user maintains their own databases containing record sets under labels organized in zones. A zone is referenced by a public-key pair. Here Alice, Bob and Carol have web servers all reachable under `www.gnu`. For Alice `www.gnu` resolves to a different address than for Bob or Carol, as their respective local name service switches (NSS) associate a user-specific public key with `.gnu`. To allow other users to resolve the names, a user's public zone information is encrypted and published in a DHT under an obfuscated query key. A user can *delegate* to another user's namespace from his local namespace to resolve foreign names. Alice can access Bob's namespace by delegating control over the name `bob` to P_{bob} in her namespace using a GNS-specific "PKEY" record. This way Alice can access Carols's webserver using the name `www.carol.bob.gnu`.

used to establish the aforementioned delegation path. Due to the use of a DHT, it is not necessary to specify the address of some system that is responsible for operating the target zone. Record validity in the DHT is established using signatures and controlled using expiration values.

12.2 Cryptography for privacy

To enable other users to look up records of a zone, all records for a given label are stored in a cryptographically signed block in the DHT. To maximize user privacy when using the DHT to look up records, both queries and replies are encrypted and replies are signed using a public key derived from the public key of the zone and the label (Figure 14). Any peer can easily validate the signature but not decrypt the reply without prior knowledge of the public key and label of the zone. Consequently, users can use passwords for labels or use public keys that are not publicly known to effectively restrict access to zone information to authorized parties.

Due to the use of a DHT, all GNS queries go to the same fully decentralized and shared global infrastructure

instead of operator-specific servers. This provides censorship-resistance and makes it impossible to target a zone-specific server because all machines in the DHT are jointly responsible for all zones — in fact, the key-value pairs do not reveal which zone they belong to. At the same time, encryption and authentication of the records is critical as it helps protect the users from effective censorship or surveillance. However, unlike the other less radical proposals to overhaul DNS, deploying GNS will be a significant challenge: GNS requires more significant changes to software, as well as a community effort to operate a DHT as a new public infrastructure.

13 Assessment

The technical approaches presented differ widely in their security goals. We summarize the key differences in Table 13.

DNS basically assumes a trustworthy IP network, the other models (except for Confidential DNS) assume that the network cannot be trusted to protect the integrity of the data. Protecting the integrity of the responses has thus been the first order of business for all approaches to secure DNS, starting with DNSSEC.

DNSSEC's limited focus means that it does not consider privacy implications of exposing requests and responses and their origin to the network. Only NameCoin and GNS try to hide the nature of client requests from the operators of the network. Here, GNS is vulnerable to a confirmation attack, so NameCoin's protection is technically stronger in terms of client request privacy. The other approaches expose the contents of the queries and replies to the operators; query name minimization (not shown) can be used to limit which servers get to learn the full query. However, clients have not assurances that query name minimization is actually deployed.

DNSSEC did try (but failed) to protect zone information against zone walks. The situation is not easily remedied by the use of stronger cryptographic primitives, as NSEC5 [19] provides an impossibility result showing that online cryptography is necessary to support NXDOMAIN responses, and preventing bulk acquisition of zone data. The proposed scheme for NSEC5 uses two different public keys to separate the offline key used to sign zone data from the online key used to generate NXDOMAIN responses. This way, compromising the online key only enables zone enumeration, but does not impact integrity. In contrast, GNS does not use online cryptography or any direct interaction with the zone's authority. GNS can store even confidential data in the name system, effectively protect it from illicit observation by the network or service operators and use offline signing, but cannot support NXDOMAIN. Finally, NameCoin deliberately made the opposite design choice and exposes the full database to all participants.

Using unsolicited DNS replies by open resolvers for traffic amplification is a well-known vector for DDoS attacks. The increased size of DNSSEC responses makes the situation worse, while caching of NSEC replies could also help reduce traffic. Some of the new approaches are not based on UDP, thus making it significantly more difficult to abuse DNS for traffic amplification.

Only the alternative approaches, Namecoin and GNS, are resistant to censorship. Approaches using traditional DNS registrars are inherently vulnerable to legal attacks where influential entities force registrars to block names.

Naturally, Table 13 falls short of considering the complete picture. For example, without padding, encrypted queries and responses may still leak information by exposting the size of the message. Also, traffic amplification is merely one vector for denial-of-service attacks, and there may be other ways to impact the fundamental security goal of availability. Our comparisson also excludes practical issues, such as the propagation delay for updates, resolution latency, and general usability.

14 Conclusion

In "Culture Is Our Business" Marshall McLuhan stated presciently:

	Protection against						Ease of
	Manipulation by MiTM	Zone walk	Client observation network	Client observation operator	Traffic Amplification	Censorship / Legal attacks	Migration / Compatibility
DNS	✗	✓	✗	✗	✗	✗	+++
DNSSEC	✓	failed	✗	✗	+/-	✗	+*
DNSCurve	✓	✓	✓	✗	✓	✗	+*
DNS-over-TLS	✓	n/a	✓	✗	✓	✗	+
Confidential DNS	✗	n/a	✓	✗	✗	✗	++
Namecoin	✓	✗	✓	✓	✓	✓	-
GNS	✓	✓	✓	✓	✓	✓	- -

*EDNS0

Table 1: Comparison of the defenses offered by the various designs and their relative deployment complexity.

> "World War III is a guerrilla information war with no division between military and civilian participation."

It appears that his prediction from 1970 remains relevant when we consider the Internet's architecture as it is woven through our everyday lives.

DNS was never designed with privacy or security as a design goal. In the battle of nation states for global dominance, any Internet infrastructure that serves a specific audience is a target for state attackers. Critical infrastructure needs to be logically decentralized and should ideally be shared globally to reduce the value of harming it. Merely encrypting DNS and Web traffic may not sufficiently reduce the effectiveness of targeted attacks against insecure designs.

Awareness exists in the DNS community that privacy is an issue, and ongoing work investigates the security, compatibility and performance implications of proposed alternatives [34]. Nevertheless, the diverse interests in the community make it virtually impossible to quickly make significant progress by consensus. Modifications to a deployed system like DNS, following the general ossification trend of the Internet, are met with inertia and usually end up with death by committee, as any significant change could not only result in serious malfunctioning, but may also impact somebody's business model or nation state interest.

The currently proposed band aids from the IETF fail to address the scope of the problem: surveillance of users, commercial censorship and the danger that DNS systems and their administrators become legitimate targets for technical, political or military attacks must be addressed better in future designs.

Acknowledgments

We thank Laura Poitras, Ludovic Courtès, Dan Bernstein, Luca Saiu and Hellekin Wolf for their help and support in preparing this report. We thank Stephane Bortzmeyer, William Aiello and the anonymous reviewers for constructive comments.

We thank the authors of NSEC5 for point out a misscharacterization of their work in a previous draft of this article.

References

[1] Why top level domains should not use wildcard resource records. `https://www.icann.org/groups/ssac/documents/sac-015-en`, 2015.

[2] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. DNS security introduction and requirements. *IETF RFC 4033*, March 2005.

[3] R. Barnes. Use Cases and Requirements for DNS-Based Authentication of Named Entities (DANE). RFC 6394 (Informational), October 2011.

[4] Jason Bau and John Mitchell. A security evaluation of dnssec with nsec3. In *NDSS*, 2010.

[5] Daniel J. Bernstein. Curve25519: new Diffie-Hellman speed records. In *In Public Key Cryptography (PKC), Springer-Verlag LNCS 3958*, 2006.

[6] Daniel J. Bernstein. DNSCurve: Usable security for DNS. `http://dnscurve.org/`, 2008.

[7] K. Bhargavan, A. Delignat-Lavaud, A. Pironti, A. Langley, and M. Ray. Transport Layer Security (TLS) Session Hash and Extended Master Secret Extension. RFC 7627 (Proposed Standard), September 2015.

[8] Internet Architecture Board. IAB statement on Internet confidentiality. `https://mailarchive.ietf.org/arch/msg/ietf-announce/ObCNmWcsFPNTIdMX5fmbuJoKFR8`, 2014.

[9] S. Bortzmeyer. Possible solutions to DNS privacy issues. `http://tools.ietf.org/html/draft-bortzmeyer-dnsop-privacy-sol-00`, December 2013.

[10] S. Bortzmeyer. DNS Privacy Considerations. RFC 7626 (Informational), August 2015.

[11] S. Bortzmeyer. DNS Query Name Minimisation to Improve Privacy. RFC 7816 (Experimental), March 2016.

[12] S. Bortzmeyer. Next step for dprive: resolver-to-auth link. `https://tools.ietf.org/html/draft-bortzmeyer-dprive-step-2-01`, July 2016.

[13] C. Contavalli, W. van der Gaast, D. Lawrence, and W. Kumari. Client Subnet in DNS Queries. RFC 7871 (Informational), May 2016.

[14] Russ Cox, Athicha Muthitacharoen, and Robert Morris. Serving dns using a peer-to-peer lookup service. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 155–165, London, UK, UK, 2002. Springer-Verlag.

[15] S. Dickinson, D. Gillmor, and T. Reddy. Authentication and (d)tls profile for dns-over-tls and dns-over-dtls. `http://tools.ietf.org/html/draft-wijngaards-dnsop-confidentialdns-02`, 2016.

[16] V. Dukhovni. Opportunistic Security: Some Protection Most of the Time. RFC 7435 (Informational), December 2014.

[17] Monika Ermert. Analyse: Usa gibt ihre wächterrolle im dns ab. `https://www.heise.de/newsticker/meldung/Analyse-USA-gibt-ihre-Waechterrolle-im-DNS-ab-3339640.html`, October 2016.

[18] Yves Eudes, Christian Grothoff, Jacob Appelbaum, Monika Ermert, Laura Poitras, and Matthias Wachs. Morecowbell - nouvelles révélations sur les pratiques de la nsa. `http://www.lemonde.fr/economie/visuel/2015/01/24/cowbells-nouvelles-revelations-sur-les-pratiques-de-la-nsa_4561547_3234.html`, 2015.

[19] Sharon Goldberg, Moni Naor, Dimitrios Papadopoulos, Leonid Reyzin, Sachin Vasant, and Asaf Ziv. Nsec5: Provably preventing dnssec zone enumeration. *IACR Cryptology ePrint Archive*, 2014:582, 2014.

[20] Amir Herzberg and Haya Shulman. Fragmentation considered poisonous: or one-domain-to-rule-them-all.org. In *CNS 2013. The Conference on Communications and Network Security*. IEEE, 2013.

[21] P. Hoffman and W.C.A. Wijngaards. Elliptic Curve Digital Signature Algorithm (DSA) for DNSSEC. RFC 6605 (Proposed Standard), April 2012.

[22] Ralph Holz. *Empirical analysis of Public Key Infrastructures and investigation of improvements*. PhD thesis, TU Munich, submitted December 2013.

[23] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman. Specification for DNS over Transport Layer Security (TLS). RFC 7858 (Proposed Standard), May 2016.

[24] Srinivas Krishnan and Fabian Monrose. Dns prefetching and its privacy implications: When good things go bad. In *Proceedings of the 3rd USENIX Conference on Large-scale Exploits and Emergent Threats: Botnets, Spyware, Worms, and More*, LEET'10, pages 10–10, Berkeley, CA, USA, 2010. USENIX Association.

[25] Allison Mankin, Duane Wessels, John Heidemann, Liang Zhu, and Zi Hu. t-DNS: DNS over TCP/TLS. `http://www.isi.edu/ant/tdns/`, 2014.

[26] Bodo Möller, Thai Duong, and Krzysztof Kotowicz. This POODLE bites: exploiting the SSL 3.0 fallback. `https://www.openssl.org/~bodo/ssl-poodle.pdf`, 2014.

[27] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. `http://bitcoin.org/bitcoin.pdf`, 2008.

[28] Anonymous (NSA). There is more than one way to quantum. `https://www.documentcloud.org/documents/1076891-there-is-more-than-one-way-to-quantum.html#document/p1`, 2014.

[29] NSA/CSS Thread Operations Center (NTOC). Bad guys are everywhere, good guys are somewhere! `http://www.spiegel.de/media/media-34757.pdf`, 2014.

[30] V. Pappas, D. Massey, A. Terzis, and L. Zhang. A comparative study of the dns design with dht-based alternatives. In *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, pages 1–13, April 2006.

[31] Venugopalan Ramasubramanian and Emin Gun Sirer. The design and implementation of a next generation name service for the internet. In *Proc. ACM SIGCOMM*, Portland, Oregon, August 2004.

[32] Redacted (NSA, S32X). QUANTUMTHEORY. `https://firstlook.org/theintercept/document/2014/03/12/nsa-gchqs-quantumtheory-hacking-tactics/`, 2014.

[33] P. Saint-Andre and J. Hodges. Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS). RFC 6125 (Proposed Standard), March 2011.

[34] Haya Shulman. Pretty bad privacy: Pitfalls of dns encryption. In *13th Workshop on Privacy in the Electronic Society*. ACM, 2014.

[35] Aaron Swartz. Squaring the triangle: Secure, decentralized, human-readable names. `http://www.aaronsw.com/weblog/squarezooko`, 2011.

[36] `http://dot-bit.org/`. The Dot-BIT project, a decentralized, open DNS system based on the Bitcoin technology. `http://dot-bit.org/`, 2013.

[37] Matthias Wachs, Martin Schanzenbach, and Christian Grothoff. A censorship-resistant, privacy-enhancing and fully decentralized name system. In *13th International Conference on Cryptology and Network Security (CANS 2014)*, pages 127–142, 2014.

[38] Nicholas Weaver. A close look at the NSA's most powerful Internet attack tool. *Wired*, 2014.

[39] W. Wijngaards. Confidential DNS. `http://tools.ietf.org/html/draft-wijngaards-dnsop-confidentialdns-02`, 2014.