2026 IEEE 11th European Symposium on Security and Privacy (EuroS&P) | 979-8-3195-1074-7/26/$31.00 ©2026 IEEE | DOI: 10.1109/EuroSP68448.2026.00094

Secrets Best Not Shared: DNS Privacy Enhancements for the Constrained IoT

Martine S. Lenders
TU Dresden
Dresden, Germany
martine.lenders@tu-dresden.de

Thomas C. Schmidt
HAW Hamburg
Hamburg, Germany
t.schmidt@haw-hamburg.de

Matthias Wählisch
TU Dresden & Barkhausen Institut
Dresden, Germany
m.waehlisch@tu-dresden.de

***Abstract*—Attackers often identify DNS traffic to disrupt or compromise Internet services. While prior work has focused on encrypting queries using DNS over TLS, HTTPS, or QUIC to counter such attacks, we consider IETF protocols designed for resource-constrained IoT devices and empirically analyze the potential of obfuscating DNS traffic in addition to encryption. We create a dataset of machine-to-machine-compatible data objects along with the corresponding DNS resolution processes, evaluating 296 deployment scenarios of resolving host names, including DNS over the Constrained Application Layer Protocol (CoAP) and an onion routing flavor of CoAP under varying link-layer conditions. We compare them to DNS over HTTPS. Using Random Forest and a header field analysis, we identify fields that leak most information. Our findings show that DNS over CoAP with equalized packet lengths, block-wise transfer, and header compression reduces the accuracy of identifying DNS frames to 86% and further to 77% with payload compression. Our approach outperforms DNS over HTTPS, where classifiers always identify DNS frames based on IP addresses. The dataset is publicly available.**

***Index Terms*—DNS over CoAP, DTLS, IoT, ML-based Classification, OSCORE, SCHC**

1. Introduction

The Domain Name System (DNS) translates names to addresses. Its original intent was to provide a "human-readable dictionary" of Internet hosts [1]. For the longest time, DNS exchanged this information without encryption. This leads to privacy risks in many deployment scenarios because it leaks sensitive information [2]–[6]. In addition, stateless DNS access exposes nodes too reflective, amplification attacks [7]–[12]. To mitigate both privacy and security risks, DNS over TLS (DoT) [13], DNS over HTTPS (DoH) [14], and DNS over QUIC (DoQ) [15] were introduced.

With the advent of the (low-end) Internet of Things (IoT) it became apparent that established technologies require reconsideration. IoT devices are constrained in memory, CPU, and energy. Most recently, we proposed DNS over the Constrained Application Protocol (CoAP) [16], [17], short DoC. DoC, like DoH, was specifically designed to hide DNS traffic within regular application packets of CoAP (as HTTPS in DoH [18]).

Providing DNS security and privacy for the IoT is extremely important. First, DNS is a fundamental network service. An attacker who blocks DNS traffic on the IoT may cause cascading outages because many IoT services such as for smart cities or smart factories are interdependent. Second, DNS leaks sensitive information that, for example, enables the identification of individual IoT devices [5], [19], which allows for targeted attacks. To counter such attacks, it is crucial to prevent eavesdroppers from identifying DNS traffic, in particular in wireless domains, which ease wiretapping. As soon as DNS traffic is identified, an attacker can interfere with DNS or apply DNS-specific analyzes. Therefore, the scope of this paper are attackers connected via wireless networks to learn about traffic (see Figure 1) but insights are not limited to this scenario.

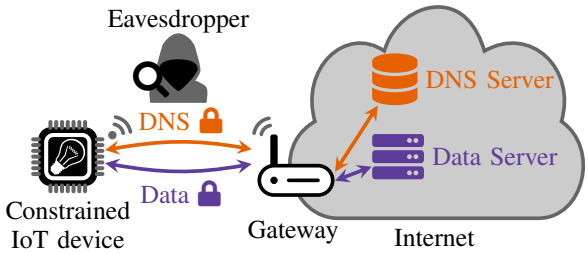


Figure 1: The attack scenario analyzed in this paper. An eavesdropper tries to distinguish data from DNS packets sent by a constrained IoT device on a wireless network.

Encryption helps to improve security and privacy but does not solve the problem completely since an attacker can derive information from meta data. Different from the core Internet, the constrained IoT deploys other means of packet modifications, which could help to hide DNS traffic in common IoT deployment scenarios [20], [21]. Static Context Header Compression (SCHC) [22] was originally introduced by the IETF to allow for IP traffic in very constrained wide-area networks. The individual header compression scheme is only identified by an opaque rule ID but allows for the elision of headers. The exchange of rules may happen out-of-band [23]–[26]. Hence, SCHC offers a transparent opportunity to omit leaking header fields. CoAP provides block-wise transfer [27], which counters fragmentation and can ensure reliable transfer on the application layer. With these mechanisms in place, header data can be obfuscated, and packet lengths be equaled out with short block sizes. Still, further compression techniques forced upon protocols may lead to even shorter DNS packets [28], which in turn may counter obfuscation.

© 2026, Martine S. Lenders. Under license to IEEE.

DOI 10.1109/EuroSP68448.2026.00094
Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

In this paper, we question whether CoAP block-wise transfer combined with SCHC header compression significantly decreases distinction of encrypted DNS and data traffic when using DoC. Additionally, we address aspects of recently proposed protocol extensions [28] that reduce the DNS packet sizes. To the best of our knowledge, this is the first paper that explores classification options of DoC within CoAP data traffic. While we use established analyzes techniques set forth in related work, this work also provides detailed insights into basic OSCORE [29] and Onion OSCORE [30]. Furthermore, we are the first to analyze the elision of header fields for privacy enhancements per peer-based SCHC rule sets. In detail, our key contributions are:

1) A comprehensive data corpus of HTTPS and CoAP traffic traces, combining data derived from the HTTP Archive [31] and corresponding DNS resource records (DoH and DoC) with 296 synthesized deployment options in unconstrained and constrained networks. This dataset does not only inform our study but may also serve as a basis for future DNS privacy research. We make this dataset publicly available. (Sections 3 and 4)
2) A careful analysis of six machine learning (ML) classifiers to assess which ML classifier serves best for further exploitation of packet header fields. We confirm that Random Forest yields the best performance. (Section 5)
3) A large scale analysis of header fields from the traffic traces using permutation feature importances of Random Forest to explore which header fields could leak DNS information to an attacker. (Section 6.2)
4) A proposal of peer-based SCHC rule sets to improve privacy by maximizing obfuscation of header information. These insights guide both more privacy-friendly DNS on the IoT and on the global Internet. (Section 6.3)

Prior work shows that DoT, DoH, and DoQ are still prone to information leakage when ML is used to distinguish DNS and data traffic [32]–[37]. They found that indicators such as inter-arrival times, header data, and packet lengths allow for fingerprinting DoH within HTTPS traffic [32]–[34], [36]. These metrics can even provide hints about the type of querying device or the requested resource. The constrained IoT with its low power, lossy link layers faces long delays and potential packet loss. This renders the obfuscation of inter-arrival times by adding artificial delays infeasible, as considered in prior work [34]. Similarly, the EDNS(0) padding option [38] applied to obfuscate DNS packet lengths places incompatible burdens as it recommends to use multiples of 128 bytes for padding [39]. This would inflate packet sizes and often enforce fragmentation, thus multiplying loss in the constrained IoT [40], which should be avoided. Following these constraints, this paper does not focus on expensive obfuscation techniques, but on the aforementioned block-wise transfer and header compression instead. Hence, we only look at packet formats.

The remainder of this paper is structured as follows. In Section 2, we provide background on the potentials of DoC and SCHC to hide DNS traffic. Section 3 describes the methods we use to generate our data corpus of traffic traces and how we execute the header field analysis. A quantitative analysis of the data corpus is presented in Section 4. In Section 5, we introduce potential machine learning models and justify our selection of Random Forest for further analysis, followed by the detailed header field analysis and our proposal of possible advancements in Section 6. We discuss related work in Section 7, and limitations and our overall findings in Section 8. In Section 9, we conclude with an outlook on future work.

2. Background

In this section, we provide background on attack scenarios, the protocols, and principles we use and evaluate in this paper.

Attacks based on distinguishing DNS traffic. Isolating encrypted DNS traffic can open a door to information leakage. An attacker can use features such as the number, timing, or length of DNS messages to infer the type of websites being queried [32], the type of device making the query [34], or even classify the type of DNS traffic as desirable or undesirable [36] in scenarios such as censorship. Once isolated, an attacker can employ the very same machine learning techniques we are using in this paper. More sophisticated machine learning techniques will only increase the accuracy of such an attack. A key defense strategy is to make DNS traffic harder to detect within overall network traffic by obfuscating it through encryption, removing identifiable fields, and standardizing packet lengths using padding or block-wise transfers. Obfuscating timing features is less of an option given the specific transmission times on the constrained IoT.

Obfuscating DNS with CoAP. We proposed DNS over CoAP (DoC) [16], [17] in the IETF to provide the benefits of DoH for the constrained IoT, *i.e.,* security and privacy of DNS traffic by encrypting and reusing an application layer protocol that is not DNS-specific. CoAP [41] is extended by two features. (*i*) Block-wise transfer to segment request and response bodies over UDP into equally-sized blocks [27] and (*ii*) Object Security for Constrained RESTful Environments (OSCORE) [29] to provide end-to-end encryption on the application layer in addition to transport encryption based on DTLS [42], [43]. Block-wise transfer has the potential to homogenize length information of DoC and regular CoAP traffic. OSCORE adds a new mode of encryption that allows protecting header and payload information even from forwarding proxies. To that end, Onion CoAP [30] has been introduced, which utilizes multiple layers of OSCORE protection to provide Onion Routing [44], [45] over several CoAP proxies. In this paper, we refer to Onion CoAP as "Onion OSCORE" to distinguish it from (D)TLS-based Onion Routing. Onion OSCORE has the potential of hiding the destination address and hostname of DoC messages from an observer and the source address from the DNS operator (*cf.,* Oblivious DoH [46]–[49]).

Eliding header fields. Static Context Header Compression (SCHC) [22] provides another option to obfuscate information in addition to encryption on the network, transport, and application layers. SCHC is a framework for header compression originally designed for Low-Power

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

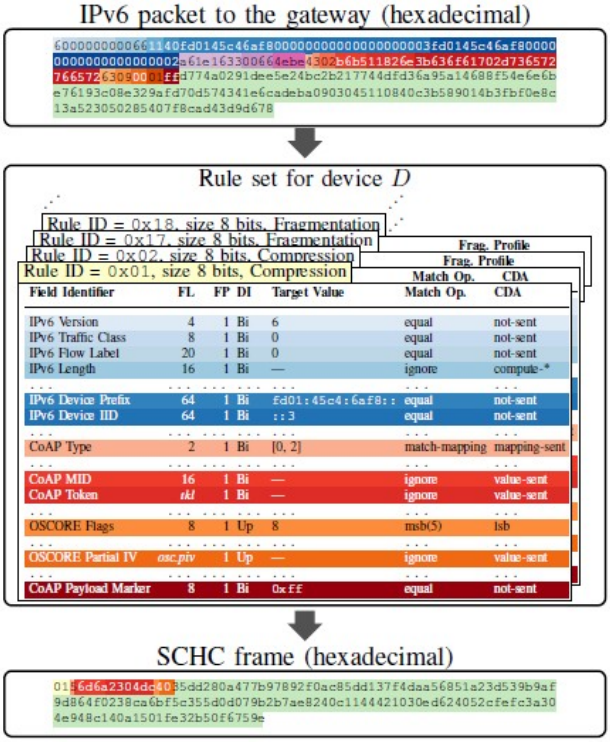


Figure 2: Static Context Header Compression (SCHC) uses rules (marked yellow) to compress and fragment IPv6 packets. Headers fields are identified by a *field identifier* and are of a certain *field length* (FL) at a *field position* (FP). If they match for a certain *direction* (DI) and a *target value* using the *match operator* (Match Op.), they compress to only a handful of bits (marked in shades of blue, purple, red, and orange) according to a *compress/decompress action* (CDA). This may cause a bit shift (see, *e.g.,* payload marked in green). The "not-sent" and "compute-*" CDAs cause a header field to be completely elided from the SCHC frame.

Wide Area Networks (LPWANs) [50], but wider deployments, *e.g.,* for 6LoWPAN or interplanetary link layers are discussed [51], [52], highlighting the relevance of this protocol in future deployments. SCHC uses rule sets—the static context—to define fragmentation and header compression of IPv6 packets on a per-device basis. These rule sets are exchanged *out-of-band*, *e.g.,* either directly burned into the firmware ROM or updated via an encrypted channel, *e.g.,* using CORECONF [23]–[26]. The rules in this rule set are identified by a rule ID, which often, together with non-compressible residues, are the only information remaining in a packet after compression when sent from one Compressor/Decompressor (C/D) to another. Other information is elided using the "not-sent" compression/decompression action (CDA), calculated implicitly using the "compute-*" CDA, or reduced to only a few bits, using CDAs such as "mapping-sent" or "lsb" (least significant bits), see Figure 2.

Since the best secrets are secrets that are not shared at all (*e.g.,* by eliding them), we see a potential for further obfuscation when using SCHC. However, a few bits and even the rule ID can become an identifier for a message flow. In this paper, we propose a *peer-based* approach to SCHC rules to obfuscate flows between two peers. The device ID that identifies rule sets usually is the link-layer address of the constrained device. Combining source and destination address to the device ID associates rule sets with a pair of peers instead of just the device. This way, we can use the same rule ID for multiple destinations and elide—rather than compress—flow information.

SCHC also provides a mechanism for fragmentation. However, since this puts additional load on the link-layer, one typically wants to avoid this whenever possible [40].

Using other DNS and data formats. The media type `application/dns+cbor` [28] is a CBOR-based alternative to the common media type `application/dns-message`. Concise Binary Object Representation (CBOR) [53] is a binary alternative to JSON. The aim of `application/dns+cbor` is to reduce DNS message sizes in constrained IoT deployments. Reducing message sizes not only helps prevent fragmentation but (unfortunately) also makes it easier to identify DoC traffic within CoAP data, similar to how DoH can be recognized within HTTPS traffic [32]–[34], [36].

3. Creating a Dataset for Privacy Analysis of DNS on the Internet of Things

In this section, we describe our threat model and how we generate the traffic for our header field analysis. The creation of our dataset is designed to gather real data where possible and combine this data with empirical measurements where real deployment is missing. Therefore, we collect data objects based on Internet-wide measurements and extend them by synthesized traffic scenarios to reflect different IoT and DNS deployments. Figure 3 summarizes our method.

3.1. Threat Model

Common IoT deployments connect constrained wireless devices to the Internet via a gateway [20]. The wireless domain poses a security challenge [21], as it creates an easy opportunity for pervasive monitoring attacks [54], *i.e.,* an adversary eavesdrops on the traffic of the IoT device (see Figure 1).

Target. The target is the user or owner of an IoT device that the client is running on.

Adversary. The adversary is an eavesdropper that collects the traffic between client and servers in a wireless domain, *e.g.,* using a radio device in promiscuous mode or a spectrum analyzer. The adversary aims to differentiate between DNS and data traffic, *e.g.,* by fingerprinting the packet, in order to examine the DNS traffic in detail.

An attacker may use consumer grade devices to generate a labeled training set a priori, similar to the approach we describe in this section. Eavesdropper outside the wireless domain are not considered. However, most obfuscation techniques discussed in this paper also apply to adversaries outside of the wireless domain. These include techniques based on header compression when the SCHC compressor/decompressor is not deployed on the gateway but closer to the servers and SCHC traffic is encapsulated [55].

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

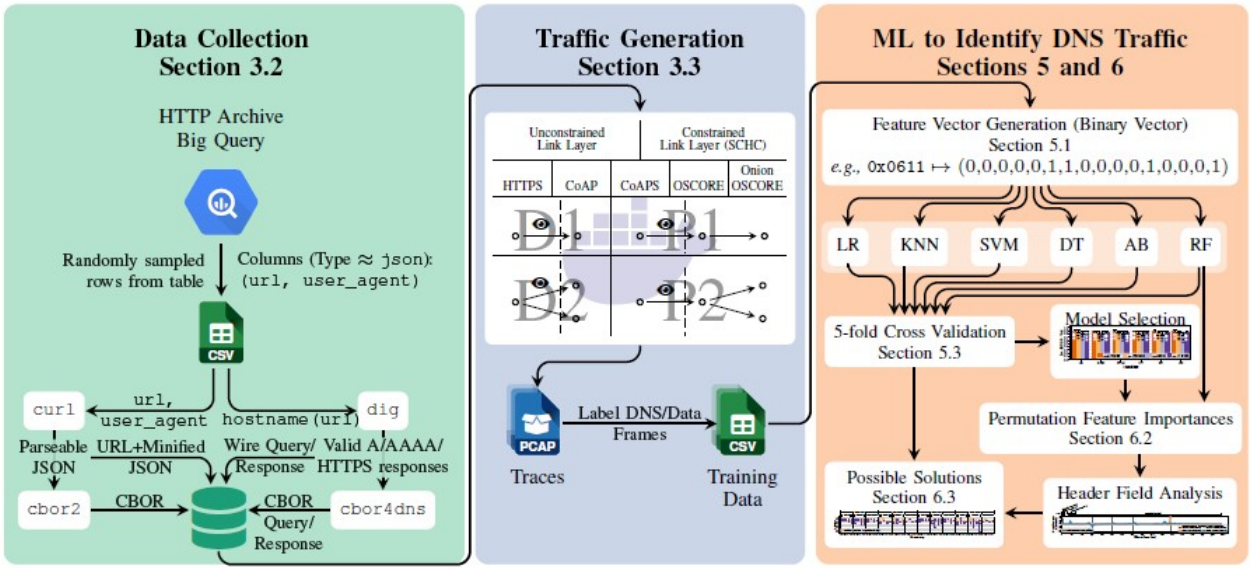


Figure 3: The method applied in this paper. We first randomly sample the HTTP Archive for data. Using that, we then generate traffic traces in a Docker-based environment. The feature vectors generated from the labeled traffic traces serve as input for cross validation of the ML models for model selection. We use RF to determine the permutation importances for every traffic scenario. These results in turn we use for our header field analysis to determine which header fields leak meta-information to an attacker. Last, we propose possible solutions based on our findings.

Threat. Prior work has shown that identifying DNS traffic, even if encrypted, and extracting its communication patterns can lead to identification of device type, the category of sites requested by the device, or the type of DNS message exchange, see Section 2. Apart from these privacy concerns, this also poses security risks as device identification allows for exploits of known vulnerabilities. Additionally, identifying DNS packets eases denial of service attacks since an attacker can disturb DNS packets on purpose to block communication.

3.2. Data Collection

Our data collection process is designed to reflect real data where possible to serve as a solid base for the evaluation of DNS privacy on the IoT. Since the constrained IoT is emerging and DNS over CoAP is currently not widely deployed, we face the challenge to approximate future deployment. The first step consists of identifying data objects that are fetched based on a hostname as part of a URL. Those objects then require a DNS resolution.

Method. Our dataset is based on HTTP Archive [31]. From that database, we fetch the URLs of JSON objects, *i.e.,* REST-based, structured machine to machine (M2M) communication data. For scalability reasons, we cannot consider all entries. Instead, to collect responses, we randomly select entries with responses of the `summary_requests` table[1]. We select the user agent and URL to gain current information, *i.e.,* requesting real responses for a month from the actual servers using curl [56]. We discard (*i*) all responses not parsable as JSON, and (*ii*) all responses larger than 1000 bytes in their minified JSON form, *i.e.,* when stripped of all non-essential whitespaces. This ensures that our dataset is based on structured data (*i.e.,* M2M-capable) and aligns with requirements of the constrained IoT (small packet sizes).

To mix-in requests that have a payload—a typical CoAP traffic pattern—and to allow for block-wise requests, we also need data objects for the requests. To that end, if a URL has more than 32 characters and has query parameters, we convert these parameters to an object. Their keys become attributes to which we assign the value of the parameter. If that object is larger than 1000 bytes as minified JSON, we discard it and the original response object to cap request sizes, too. From those URLs, we remove the query parameters.

Then, we query the A, AAAA, and HTTPS DNS records of the URL hostnames using a public resolver We collect HTTPS records since they also contain IP information [57]. This reflects future deployment where DoC/CoAP-specific SVCB records [58] guide DoC and CoAP requests. To ensure repeatability and explainability of our results, we pre-collect these records and do not request them live during our traffic generation.

Setup. In November 2024, we randomly picked 6,242,477 entries from the HTTP Archive. Of those, we use the recorded pairs of the URL and user agent to request the real responses for a full month, February 2025. This leads to 58,768 URLs with request and response objects. We gain DNS data by resolving hostnames from a European vantage point via the public DNS resolver 9.9.9.9. We store the JSON object responses, including the URL, the converted query parameters, the HTTP status code that curl reported, and all successful DNS query and response messages.

We convert the collected JSON objects into CBOR using the `cbor2` library [59] and convert the DNS messages to `application/dns+cbor` using `cbor4dns` [60], which implemented November 2024 version of CBOR DNS [61] at the time of conversion.

1. The schema of this database changed during our study. This information now resides in the `crawl` table, but the information is still there.

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

TABLE 1: Data and DNS format dataset combinations.

Data	DNS	Short Name
`application/json`	`application/dns-message`	JSON & DNS
`application/cbor`	`application/dns-message`	CBOR & DNS
`application/json`	`application/dns+cbor`	JSON & CBOR
`application/cbor`	`application/dns+cbor`	CBOR & CBOR

When combining each data and DNS format, our measurements provide four datasets of traffic generation, see Table 1.

3.3. Traffic Generation

Having a pre-collected set of data objects and principle DNS payloads at hand, we now need to create traffic traces that reflect how a client interacts with DNS and data servers. These traces will serve as input to reproducibly emulate what an eavesdropper could observe and analyze. This section describes the details of the different scenarios.

Network topologies. We create traffic for four scenarios to provide a comprehensive view on common IoT deployments [20], [21] All scenarios are visualized in Figure 4. The *client* represents a constrained IoT device, the *dashed lines* a gateway, similar to the principle setup in Figure 1. Each proxy is located between a gateway and the servers to offer an additional level of obfuscation depending on the DNS transport. When using SCHC, the upstream SCHC compressor/decompressor can either run on the gateway or between the gateway and the servers or the proxy.

1) *D1 (direct, single server)*, Figure 4(a): One shared server that serves both DNS and non-DNS data. The client communicates to directly with the server.
2) *P1 (via proxy, single server)*, Figure 4(b): One shared server that serves both DNS and non-DNS data. The client communicates to the server via proxy.
3) *D2 (direct, split server)*, Figure 4(c): Two servers that serve DNS and non-DNS data. The client communicates to both servers directly.
4) *P2 (split server, via proxy)*, Figure 4(d): Two servers that serve DNS and non-DNS data. The client communicates to both servers via a proxy.

DoC was designed for constrained networks, *i.e.,* networks with low throughput, high packet loss, asymmetric links, and potential for fragmentation. We assume these properties on the wireless link connecting the client to the Internet, monitored by the eavesdropper.

DNS protocols. To compare different security modes of DoC with DoH, we generate DoC traffic via DTLS (*i.e., CoAPS*) and with *OSCORE* to secure messages. Additionally, we use *Onion OSCORE* [30] with randomized CoAP MIDs and tokens. For both DTLS and OSCORE, we use pre-shared keys (PSKs) in AES128 encryption. For baseline comparison, we run DoC using unencrypted *CoAP*.

To test DoH, we rely on HTTP/2 for two reasons. First, HTTP/2 is more prevalent than HTTP/3. Second, using HTTP/2 instead of HTTP/3 allows us to gain insights into TCP effects on the classification of DNS and data traffic.

The DTLS and OSCORE libraries prefer AES128 PSKs, and the DTLS library only implements DTLS v1.2 at the time of evaluation. To compare DoC and DoH fairly, we use TLS-PSK with TLS v1.2 and AES128 in DoH scenarios. Both DoC and DoH servers, as well as the data servers, operate on their default port (443 for HTTPS, 5683 for CoAP and OSCORE, 5684 for CoAPS). For HTTPS, CoAPS, and Onion OSCORE, the proxy can obfuscate the intended target of requests and the origin of the server responses. In fact, the effects of Onion OSCORE are only relevant when used in conjunction with a proxy, even in the P1 scenarios.

Details on resolving DNS and fetching data, and payload labeling. We resolve the hostname carried in the URL by querying a DNS server. We prioritize HTTPS over AAAA and AAAA over A records, depending on availability of data (see Section 3.2). For DoC, we use FETCH requests [16], for DoH POST. The client does not cache the response, emulating IoT devices that query names at start-up, *e.g.,* for self-configuration, or after long sleep cycles when DNS response TTLs are expired [34].

After resolving a hostname, the client sends a request to the data server. If a request object is available, the client serializes it to *CBOR* or minified *JSON*, depending on the scenario. If no request object is available, the client uses GET.

Including a real URL hostname in the unprotected part reduces the efficiency of SCHC rule definitions. To cover this, the client uses the placeholder hostnames "dns-server" and "coap-server" (or "http-server") in the CoAP `Uri-Host` option (or HTTP `Host` header). These replacements *only* happen in CoAP or HTTP headers to simplify SCHC rule deployment. For the hostname resolution above, we still use the original URL hostname. The server serializes the response object as *CBOR* or minified *JSON*, depending on the experiment. It uses the HTTP Status Code or a corresponding CoAP response code, which we collected before. We then strip the link-layer headers since this information can be randomized [62].

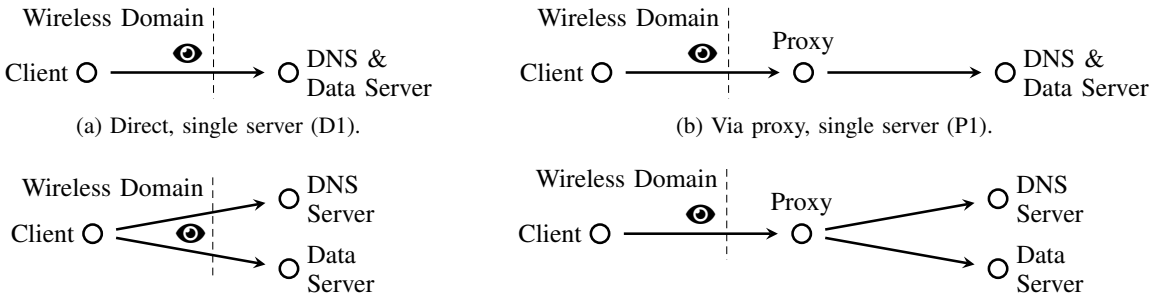
We label the link-layer payloads as *DNS frame* or *data message*, depending on whether the fragment, segment, or block belongs to a DNS or data message. For binary classification, we discard all frames not belonging to either category (TCP / (D)TLS handshake messages, empty CoAP ACKs, *etc.*).

Block-wise transfer. Our payloads have a size of $\leq$ 1000 bytes. When evaluating block-wise transfer in CoAP, CoAPS, OSCORE, and Onion OSCORE scenarios, we specify a block size of 64 bytes. For (Onion) OSCORE, we put the block options in the protected part of the message.

For non-block-wise scenarios, we leave the default block size value of the CoAP library (1024 bytes).

Link layer. We send the packets either uncompressed and unfragmented, *i.e.,* over an *unconstrained* link layer, or compressed and fragmented using *SCHC* with a rule ID of length 8 bit, *i.e.,* simulating a 127 bytes PDU *constrained* link layer. In case of the unconstrained link layer, we use Ethernet. For SCHC, we craft compression rules for each protocol and network architecture. We only evaluate the CoAPS, OSCORE, and Onion OSCORE scenarios here. For the split server scenarios (D2 and some P2), there are

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.



(a) Direct, single server (D1).

(b) Via proxy, single server (P1).

(c) Split server, direct (D2).

(d) Split server, via proxy (P2).

Figure 4: Four setups how a client requests DNS and data. We use these for traffic generation. 👁 marks the eavesdropper.

multiple possibilities to configure the SCHC compression rules:

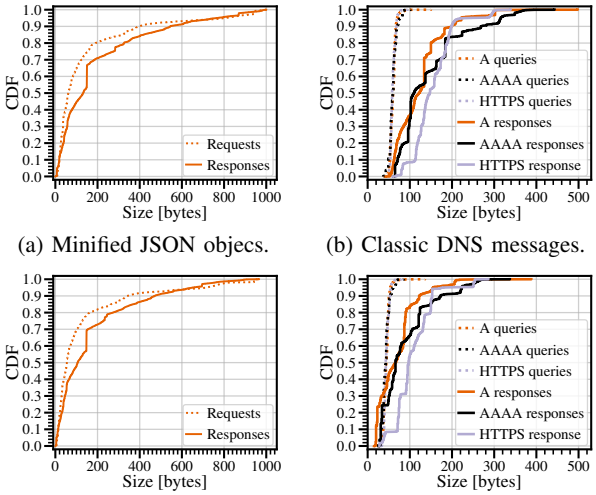
1) *Split rules* (short *split*): One rule for each server.
2) *Minimum rules* (short *min.*): One rule for both servers and use the "mapping-sent" CDA to distinguish between the differences in the headers.
3) *Peer-based rules* (short *peer*): One rule *set* for each server. This uses a function of information from both peers as device ID. For our purposes, this function XORs the link-layer source and destination addresses of the peers and assumes both peers know these addresses of each other, *e.g.,* via CORECONF [24]–[26].

In the single server scenarios (D1 and P1, but also most P2), we use a single rule to compress data and DNS frames. This elides and reduces header fields as much as possible. The protected part of OSCORE stays uncompressed [63], [64] since OSCORE libraries rarely support this feature.

To compress CoAPS frames, we adapt a proposal by Fragkiadakis [65] to comply with our scenarios. Control messages, such as empty ACKs or handshake messages, have their separate compression rules.

To implement fragmentation, we use No-ACK mode with datagram tag size 2, fragment compressed number field size 1, and reassembly check sequence (RCS) field size 4, using CRC32 for RCS. Together with the 8-bit rule ID this results in a fragmentation header size of 11 bits for regular fragments and 15 bits for the final All-1 fragment.

Total deployment scenarios. Combining all possible configurations (*i.e.,* deploying all five protocols in all four network setups, considering two block sizes and all SCHC rules), we create 296 deployment scenarios, see Table 2. In the proxy scenarios of CoAPS over SCHC and in Onion OSCORE, server identifiers are not visible in the unencrypted headers. Therefore, the *min.* and *peer* rules apply only in D2 scenarios. For plain OSCORE, the *min.* scenario applies in P2 scenarios because the `Uri-Host` option, which contains the server name, is unprotected. Consequently, we have the following number of variations. In CoAPS, 3 variants for each D2 scenario (all 3 SCHC variants), in OSCORE 6 variants for each D2 scenario (both OSCORE variants and all 3 SCHC variants) and 3 for each P2 scenario (both OSCORE variants but 2 SCHC variants only for plain OSCORE).



(a) Minified JSON objecs.

(b) Classic DNS messages.

(c) CBOR objects.

(d) DNS CBOR messages.

Figure 5: Size distributions of differently encoded data objects and DNS messages. Note that x-axis scaling differs for data objects and DNS messages.

4. Overview of Data Corpus

In this section, we provide an overview of the data corpus that we created (see Section 3), which we make publicly available. Our set of data objects contains 58,768 request and response objects each. 8,541 URL hostnames for these object pairs resolve to HTTPS records. 18,543 hostnames resolve to AAAA records. 10,691 hostnames have an AAAA records but no HTTPS record and 39,533 hostnames exclusively resolve to A records. All hostnames with an HTTPS record also have a corresponding AAAA record and all names with an AAAA record also have an A record.

Figure 5 shows the CDFs for the data objects and DNS message sizes. 80% of JSON request objects are 189 bytes long or shorter, while 80% of CBOR request objects are 165 bytes long or shorter. 80% of JSON response objects are 326 bytes long or shorter, while 80% of CBOR response objects are 274 bytes long or shorter. 99% of classic DNS queries are smaller than 83 bytes and 150 bytes max. The responses can be up to 498 bytes long. CBOR-based DNS queries are a maximum of 136 bytes long. Their responses can be up to 388 bytes long. Both CBOR-based message types are shorter than classic DNS

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

TABLE 2: Overview of all scenarios for traffic generation. Z is the block size. The numbers in each cell count the number of configurations per protocol as outlined in Section 3.3.

Formats		Network Scenario																												Sum
		Unconstrained Link Layer																Constrained Link Layer (SCHC) (3 variants if applicable)													
		HTTPS				CoAP(S)								(Onion) OSCORE								CoAPS								(Onion) OSCORE							
						$Z = 1024$				$Z = 64$				$Z = 1024$				$Z = 64$				$Z = 1024$				$Z = 64$				$Z = 1024$				$Z = 64$			
		D		P		D		P		D		P		D		P		D		P		D		P		D		P		D		P		D		P	
Data	DNS	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	
JSON	DNS	1	1	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	3	1	1	1	3	1	1	2	6	2	3	2	6	2	3	74
	CBOR	1	1	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	3	1	1	1	3	1	1	2	6	2	3	2	6	2	3	74
CBOR	DNS	1	1	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	3	1	1	1	3	1	1	2	6	2	3	2	6	2	3	74
	CBOR	1	1	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	3	1	1	1	3	1	1	2	6	2	3	2	6	2	3	74
	Sum	4	4	4	4	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	4	12	4	4	4	12	4	4	8	24	8	12	8	24	8	12	296

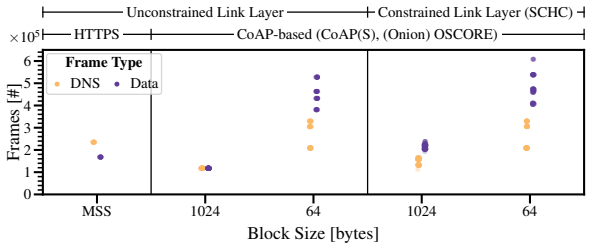


Figure 6: The number of frames per scenario, grouped by protocol and link layer (top), and block size (bottom). In HTTPS, we use the Maximum Segment Size.

or JSON. DNS messages encoded in CBOR are also much shorter than even CBOR-serialized data objects.

The number of frames generated in each scenario is shown in Figure 6. In the scatter plot, each data point represents one scenario. They are grouped by protocol and link layer (top x-axis), and block size configuration (bottom x-axis). The HTTP/2 library splits headers and payloads of each stream into two separate TCP exchanges, causing HTTPS to have approximately double the number of frames than unconstrained CoAP with block size 1024 bytes in all security modes. Naturally, more frames are generated with SCHC fragmentation and CoAP with smaller block sizes.

The lengths of the frames generated in each scenario are shown in Figure 7 as scatter plots similar to Figure 6, separated by mean (Figure 7(a)) as well as minimum and maximum (Figure 7(b)). In general, DNS frames are shorter than data frames, due to their payload lengths (see Figure 5). Even with block size 64 bytes, data frames reach sizes of over 1 kByte in unconstrained scenarios. Unusually long URI paths in the CoAP header are the cause of these larger frame sizes. Only the body is split into blocks, options in unprotected headers are not segmented. Undesired SCHC fragmentation *together* with a block size of 64 bytes leads to similar length distributions between DNS and data frames.

5. Machine Learning to Identify DNS Traffic

We use established machine learning techniques and supervised learning algorithms to create an algorithmic foundation for our header field analysis. In the following, we discuss all necessary steps, which are also visualized in Figure 3.

5.1. Feature Vector Generation

First, we generate the feature vector for 5-fold cross validation and permutation feature importances. We only regard packet formatting because prior observations render obfuscation techniques to manipulate packet timings infeasible due to restrictions put onto the sender by the constrained link [34]. Due to our emulated network scenarios, timings also do not yield realistic observations. Our goal, after all, is to algorithmically determine leaking header field information. Since our dataset is very heterogeneous, the entire packet is needed for cross validation and determining feature importances. SCHC operates on the bit level. Therefore, we translate the bytes of the observed Ethernet payloads and SCHC frames into *binary vectors*, *i.e.,* each byte B_i is translated to an 8-dimensional vector, where the bits $b_{i,j}$ of B_i are the components of the vector.

$$B_i = \sum_{j=0}^{7} b_{i,j} 2^j \mapsto (b_{i,j})_{j=0}^{7} \qquad b_{i,j} \in \{0,1\}$$

We concatenate the resulting array of vectors to one large vector.

$$[B_i]_{i=0}^{n} = \left[(b_{i,j})_{j=0}^{7}\right]_{i=0}^{n}$$
$$[B_i]_{i=0}^{n} \mapsto (b_{0,1},\ldots,b_{0,7},\ldots,b_{n,1},\ldots,b_{n,7})$$

Input vectors need to be real number vectors of the same length to be processed by classification models. We pad vectors of smaller dimension with "2". Since all vectors used to be packets at byte level, the number of 2s added to each vector is a multiple of 8.

$$(b_{0,1},\ldots,b_{n,7}) \mapsto (b_{0,1},\ldots,b_{n,7},\ldots,2,2,2,2,2,2,2,2)$$

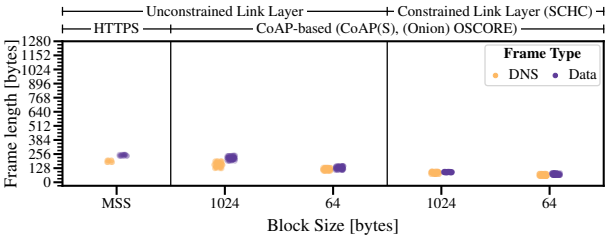
We then use Min-Max-Normalization [66] to scale the components of all vectors X to a range between 0 and 1, *i.e.,* 0 remains 0, 1 becomes 0.5, and 2 becomes 1.

$$X_{\text{scaled}} = \frac{X - \min(X)}{\max(X) - \min(X)}$$

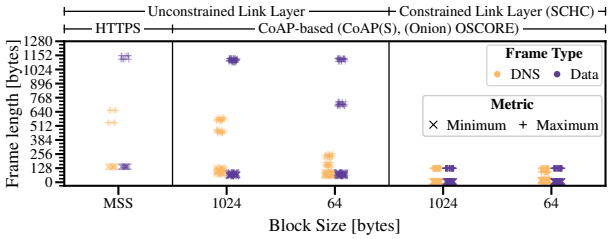
5.2. Possible Machine Learning Models

We base our analysis on common classifiers used in related work [32]–[34], [67], [68], namely Logistic

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.



(a) Mean lengths



(b) Minimum and maximum lengths

Figure 7: The mean, minimum, and maximum lengths of frames per scenario, grouped by protocol and link layer (top x-axis), and block size (bottom x-axis). In HTTPS, we use the Maximum Segment Size.

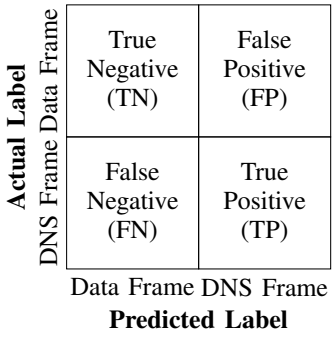


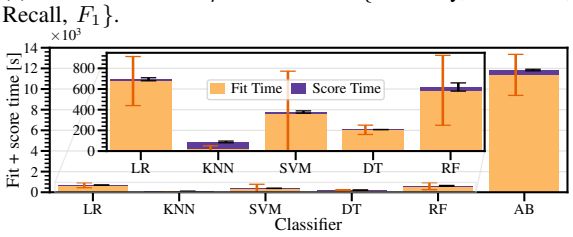
Figure 8: Confusion matrix of our binary traffic classification. Data frames are frames that belong to a data request or response, DNS frames are frames that belong to a DNS request or response.

Regression (*LR*), k-Nearest Neighbors (*KNN*), Support Vector Machine (*SVM*), Decision Tree (*DT*), Random Forest (*RF*), and AdaBoost (*AB*). We also considered Naïve Bayes (*NB*), but initial tests showed underwhelming performance in our scenarios. We explicitly do not look into deep learning classifiers, as most prior work on DoH only used classic feature-based classifiers as well [33], [34], [36] and we want to (*i*) stay comparable with these prior work and (*ii*) use the features to be able to interpret and explain leaking information. To compare the models, we use 5-fold cross validation with *scikit-learn* [69], or—for all but DT and AB—the CUDA-based alternative *cuML* [70]. We use most default parameters of *scikit-learn*/*cuML*, *i.e.,* no fixed random state and 5 unshuffled, stratified folds for 58,768 DNS and data message pairs, each, on the application layer—even more when considering fragmentation and block-wise transfer. For LR, we perform a maximum of 5000 iterations, for KNN we use brute-force search to compute the nearest neighbors, for SVM we use $C = 0.01$, for RF we use 250 trees with a maximum depth of 9 in the forest, and for AB we use 250 estimators. These numbers represent a rough average of the values used in related work [32]–[34], [67], [68].

5.3. Model Selection

In our machine learning analysis, we classify DNS frames as "positive" and data frames as "negative" (see the confusion matrix in Figure 8). We perform 5-fold cross validation for all classifiers and determine accuracy, precision, recall, and F_1 score.

(a) Mean of means $\bar{\mu}_m$ for $m \in \{\text{Accuracy, Precision, Recall, } F_1\}$.



(b) Fit & score times.

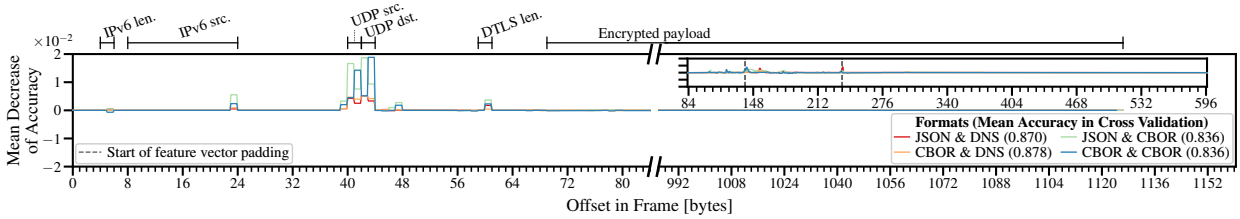
Figure 9: The mean of the means of the accuracy, precision, recall, and F_1 score ($\bar{\mu}_m$) over the 5-fold cross validation of all 296 scenarios as well as mean fit and scoring times. This backs our decision to use RF as a classifier. The scoring times also include scoring for *balanced accuracy* and the *area under the receiver operating characteristic curve (ROC AUC)* which we did not include in our final analysis for space.

In our header field analysis, we use the means $\bar{\mu}_m$ and $\bar{\sigma}_m$ of mean $\mu_{m,s}$ and standard deviation $\sigma_{m,s}$ for all metrics $m_{s,f}$ of the 296 scenarios s for each fold f, **mean** and **std** being the functions to calculate the mean and standard deviation of a dataset.

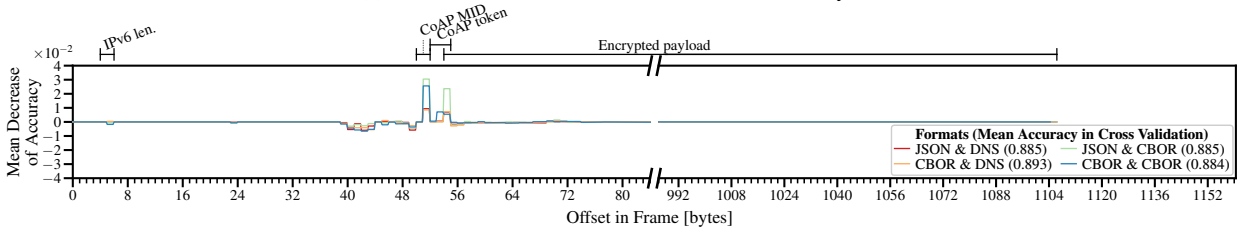
$$
\begin{aligned}
m &\in \{\text{Accuracy}, \text{Precision}, \text{Recall}, F_1\} \\
s &\in \{1, \ldots, 296\} \\
\mu_{s,m} &= \mathbf{mean}\left(\{m_{s,f}\}_{f=0}^{5}\right) \\
\sigma_{s,m} &= \mathbf{std}\left(\{m_{s,f}\}_{f=0}^{5}\right) \\
\bar{\mu}_m &= \mathbf{mean}\left(\{\mu_{s,m}\}_{s=1}^{296}\right) \\
\bar{\sigma}_m &= \mathbf{mean}\left(\{\sigma_{s,m}\}_{s=1}^{296}\right)
\end{aligned}
$$

We provide these means of mean accuracy, precision, recall, and F_1 in Figure 9. Relatively, all classifiers perform equally well in all scenarios, *i.e.,* there is no classifier that has, *e.g.,* higher accuracy for a certain group of scenarios (not shown). However, in 9.6% of all scenarios, SVM only completed 2–3× of 5 folds due to memory exhaustion (not shown). We exclude it from

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.



(a) Unconstrained, P2, CoAPS, block size 1024 bytes



(b) Unconstrained, D1, OSCORE, block size 1024 bytes

Figure 10: Example results of unconstrained scenarios without block-wise transfer where length only has little permutation importance. Note the broken x-axes and different y-axis scalings.

further consideration. AB needs one order of magnitude more time than the other classifiers, see Figure 9(b). We exclude it as well. Both AB and DT are not accelerated with cuML, but DT performs relatively well. LR has the best mean accuracy and F_1 score, closely followed by RF (see Figure 9(a)). RF, on the other hand, is more than one minute faster per fold on average. DT and KNN are very comparable in their results, and while being the fastest, they perform the worst relatively when it comes to accuracy and F_1 score. Additionally, DT tends to overfitting in case of large datasets, which RF solves. In total, RF is a good compromise between best results and runtime. Other works also use RF to identify encrypted DNS traffic [32]–[34], [67], which bolsters our decision. We present more detailed results of RF in Appendix A.

6. Evaluation

In this section, we present the results of our ML-driven analysis to distinguish non-DNS and DNS traffic and present details on the ML configuration.

6.1. Configuration

First, we train the RF classifier on 80% of each dataset. For the remaining 20%, we test the significance of each bit or byte. To that end, we use *permutation feature importances* [71] as mean decrease in accuracy with two permutation repeats. We initialize both classifier and the permutation importances function with the same random seed. This ensures that a potential change in accuracy is based on the permutation of the feature vector. We focus on the accuracy for two reasons. (*i*) Our dataset is mostly balanced, and (*ii*) we account for true negatives, *i.e.,* correctly identified data frames. The advantages of the F_1 score, precision, and recall do not hold in our case. It is also worth noting that the F_1 score is proportional to the accuracy (see also Appendix A).

To speed up calculations in unconstrained scenarios, only the byte level can be considered. To account for this, we modified the *scikit-learn* function to permute entire 8-column groups (*i.e.,* bytes) for these scenarios, instead of using the usual single column (*i.e.,* bits). Based on that, we map the bits or bytes to the corresponding header fields or compression residues, respectively, to get insights into the importance of a (compressed) header field.

6.2. Finding Leaks Using Feature Importances

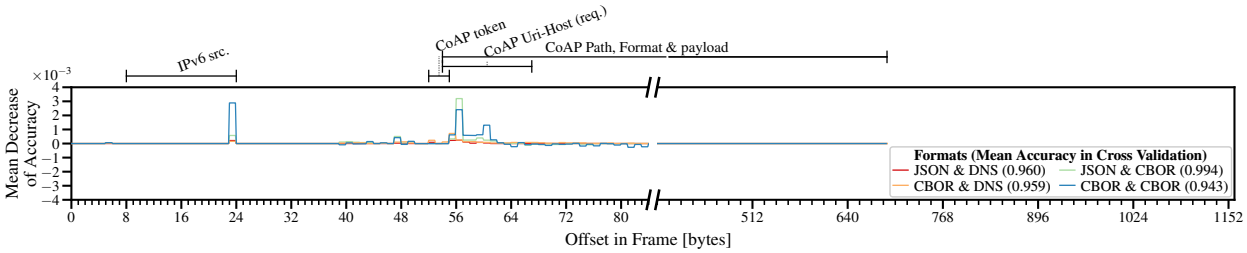
Showing all scenarios would lead to 74 plots (294 scenarios ÷ 4 format combinations). We present only a selection of plots that summarizes our findings. Please find the full set of plots in our artifacts.

Each plot shows the mean decrease of accuracy for each byte or bit, highlighting the affected header fields on the top x-axis. The overall mean accuracy from cross validation is noted in the legend. We do not align the y-axes, *i.e.,* each y-axis is cut at the largest absolute change, because for permutation importance, it is more important that there *is* a difference in score after shuffling a column. For visibility reasons, we have to cut the x-axis. If there are relevant changes in this part, we embed a rescaled zoom. Exceptions where we show the full plot are scenarios with a lot of important columns containing padding of the feature vector (see Section 5). In any case, groups of very important padding columns are marked with a dashed line. These correlate with the length of the frame. We now summarize our findings.

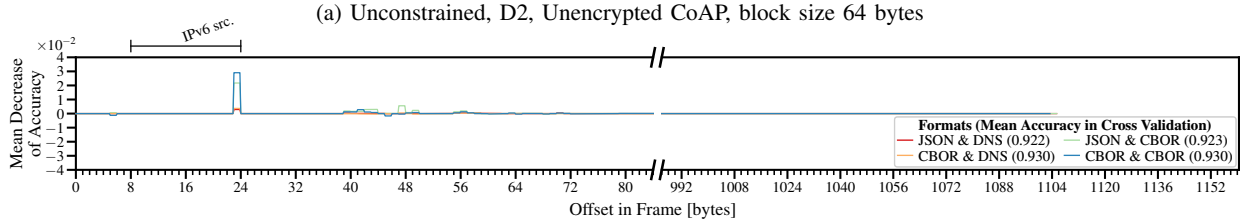
Length may not be that important. When considering the entire packet, the length of the packet is usually of low importance to distinguish frames. We provide two of those examples in Figure 10. Rather, destination information (Figure 10(a)) or monotonically growing counters, *e.g.,* non-randomized CoAP MIDs (Figure 10(b)), have higher importance.

Leak 1: Destination information. Source and destination information can have the most importance, especially when clients communicate with different DNS and data servers directly (D2). Figure 11 shows some examples of bits of the source or destination address having by far the largest importance to RF, even when packets are completely unencrypted (Figure 11(a)) or when compressed to a single bit with SCHC minimum rules (Figure 11(c)). In

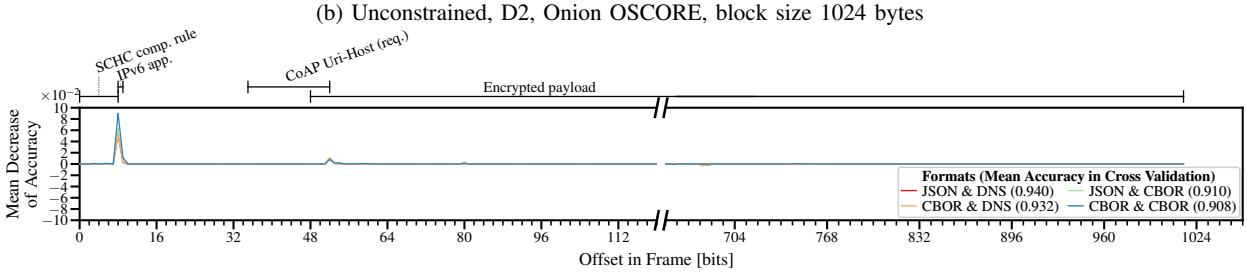
Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.



(a) Unconstrained, D2, Unencrypted CoAP, block size 64 bytes

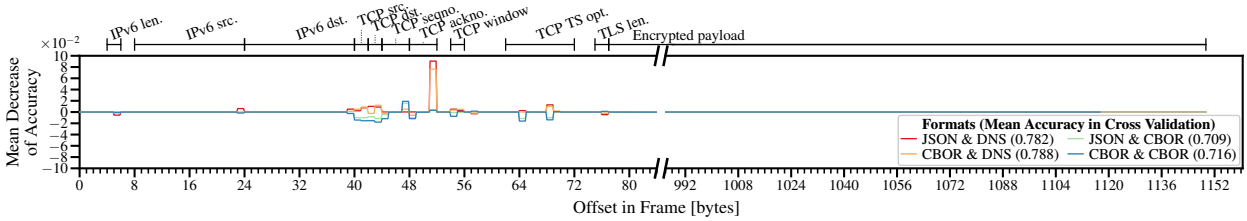


(b) Unconstrained, D2, Onion OSCORE, block size 1024 bytes

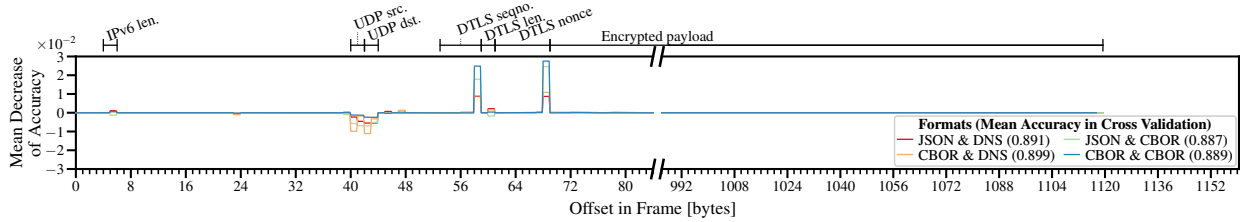


(c) SCHC, *min.*, D2, Onion OSCORE, block size 64 bytes

Figure 11: Example results of permutation importances for D2 scenarios leaking source or destination information. Note the broken and differently scaled x-axes for unconstrained, block size, or SCHC and different y-axis scalings.



(a) Unconstrained, P1, HTTPS



(b) Unconstrained, D1, CoAPS, block size 1024 bytes

Figure 12: Example results of scenarios where monotonically growing counters had the largest permutation importances. Note the broken x-axes and different y-axis scalings.

Figure 13: Violin plots of the accuracy of RF for 5-fold cross validation for block size 1024 bytes vs. block size 64 bytes of unconstrained scenarios. Horizontal bars represent minimum, mean, and maximum of all 5 folds from bottom to top.

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

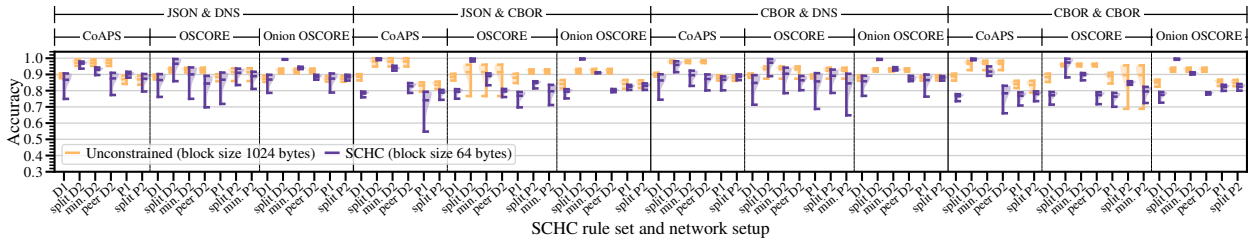


Figure 14: Violin plots of the accuracy of RF for 5-fold cross validation for unconstrained block size 1024 bytes vs. SCHC block size 64 bytes scenarios. Horizontal bars represent minimum, mean, and maximum from bottom to top.

addition to addresses, unencrypted hostname information (Figure 11(a)) and transport layer ports can leak information when different ports are used for DNS and data services, even through a proxy (Figure 10(a)).

Leak 2: Alternating DNS and data message patterns. In most IoT scenarios we consider in our setup, DNS query-response pairs precede a data exchange [34]. This allows the classifier to recognize this pattern based on monotonically growing counters, such as TCP sequence and acknowledgment numbers, DTLS sequence numbers, CoAP MIDs, CoAP tokens, and even carelessly constructed cipher nonces. Figure 12 shows a selection of scenarios where the permutation importance of these features is especially pronounced. Figure 10(b) shows this also for the OSCORE case. The DTLS explicit cipher nonce is leaking information (see Figure 12(b)). This is due to TinyDTLS, a common library we also use in our evaluation. TinyDTLS copies the epoch (always 1 for encrypted application data) and sequence number fields for usage as cipher nonce. This results in the same importance as the sequence number.

6.3. Analysis of Possible Solutions

Beyond already known solutions, *e.g.,* binding to randomized source ports [72], we propose the following steps to increase obfuscation of DNS traffic and close the leaks seen in Section 6.2.

Solution 1: Block-wise transfer. Even though length has little importance in most scenarios, we can gain small reductions of mean accuracy when using a small block size of 64 bytes with CoAP. Figure 13 compares the accuracy of RF from our 5-fold cross validation as violin plot for block sizes of 1024 bytes and 64 bytes for all unconstrained and encrypted CoAP scenarios grouped by format, protocol, and network setup. The horizontal bars show minimum, mean, and maximum from bottom to top. For plain OSCORE in general, but also for proxy-based or split server approaches based on other protocols, we see large reductions of minimum accuracy, sometimes over 30%.

Solution 2: Peer-based SCHC rules. Using peer-based SCHC rules, we can further reduce accuracy by obfuscating destination information and close leak 1, see Section 6.2. Figure 14 compares the accuracy of RF from our 5-fold cross validation as violin plot for unconstrained scenarios with block size 1024 bytes, *i.e.,* the same values as in Figure 13, and SCHC scenarios with 64 bytes as for all encrypted CoAP scenarios. If there are different rule sets for SCHC, we just compare with the same unconstrained setup. Combining block-wise transfer and SCHC, we can reduce the accuracy in most scenarios, except *split* D2. With *peer* D2 we gain similar results as for P2, even with plain OSCORE, removing the necessity of the overhead of Onion OSCORE there. However, the length, or rather the padding of the feature vector, now becomes an important feature again (see Figure 15). Nevertheless, the accuracy is only 77%-86% (depending on DNS and data format) compared to 91%-100% in unconstrained scenarios, including HTTPS (not shown, see Appendix A). This leak comes from the last block, which is not equalized in length.

Solution 3: Onion OSCORE with randomized IDs via proxy. We can reduce accuracy to a minimum when randomizing the CoAP MID and tokens as well as using Onion OSCORE via proxy compared to plain OSCORE, see Figures 13 and 14. This obfuscates alternating DNS and data message patterns and closes leak 2, see Section 6.2. Looking at feature importances in Figure 16, destination information but also the OSCORE Partial IV as a remaining counter help to identify DNS frames. Since the Partial IV is used for replay window detection, it can not be randomized. However, there are IETF discussions to encrypt it [73]. In Figure 13, we actually see an *increase* in accuracy, especially in the minimum, with Onion OSCORE compared to plain OSCORE, when the `application/dns+cbor` format is used, showing again that length here becomes more important. Similar effects are visible with SCHC in Figure 14. In both cases, a combination of more prominent destination information (in unconstrained scenarios, ephemeral ports are used), size, and the singular Partial IV, make it easier for RF to successfully classify. Randomizing the Partial IV in the traces to simulate its encryption reduces the accuracy to under 75%, even 65% in certain cases (not shown).

7. Related Work

To the best of our knowledge, this is the first paper to analyze (*i*) how well DoC traffic can be distinguished from CoAP data traffic and (*ii*) the potentials of header compression to omit information to further improve obfuscation. In this section, we discuss related work in detail.

Privacy and security concerns related to DNS on the IoT. Improving DNS security and privacy is an ongoing topic in Internet standardization [74], [75]. In particular, protection against pervasive monitoring is important since this is considered an attack [54]. Very recent discussions [76] about DNS security and privacy guidelines [72], [77] for the IoT recommend additional obfuscation of header fields, such as the Server Name Indicator (SNI) in TLS with DoH, DoQ, or DoT using Encrypted Client Hello [78]. Our proposal to use Onion OSCORE to obfuscate hostname information within the CoAP header is

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

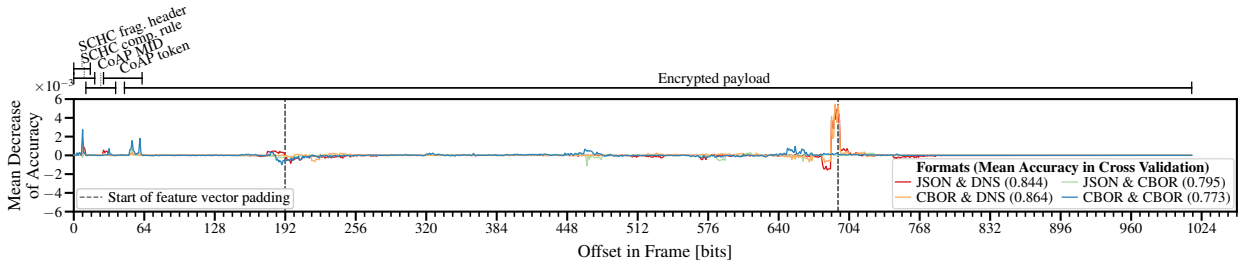
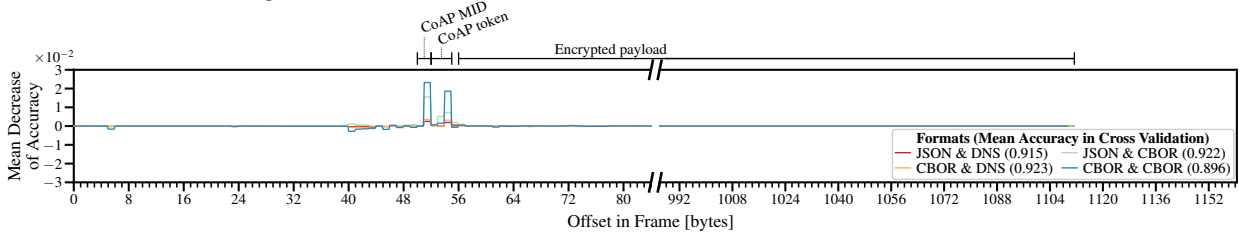
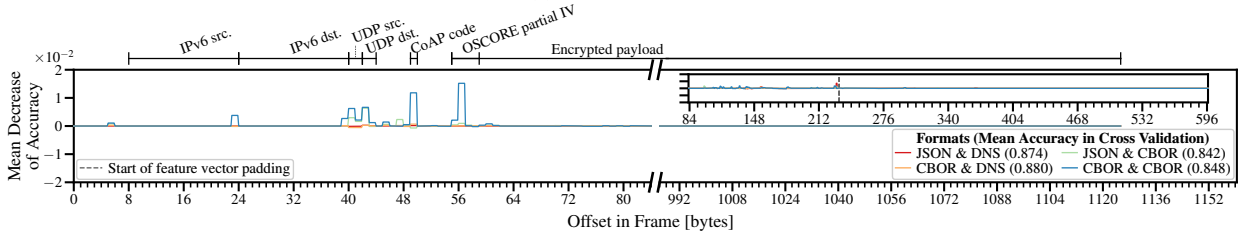


Figure 15: Permutation importances for SCHC D2 scenario with plain OSCORE with block size 64 bytes leaking information due to its lengths.



(a) Plain OSCORE.



(b) Onion OSCORE.

Figure 16: Permutation importances for unconstrained P2 scenario with and plain OSCORE and Onion OSCORE with block size 1024 bytes. Note the broken x-axes. While with plain OSCORE the MID and token are the most important features, their importance fades due to randomization. Partial IV and destination information become more prominent.

in line with these recent suggestions, but has not yet been analyzed.

Traffic anonymization. Eliding header fields is an anonymization technique, since removing headers does not necessarily offer privacy [79]. Most approaches to anonymize traffic, such as Tor [45], LAP [80], or HORNET [81], are based on encryption and implement obfuscation of the sender [82] and "hiding in the crowd" [83] approaches. Hiding metadata of the transport layer by encrypting transport header fields is often proposed [84] to support privacy and to counter network ossification due to middleboxes. Beyond privacy, RFC 8701 [85] discusses randomization of reserved header fields to counter network ossification. All these approaches still carry (even opaque) metadata. Completely omitting data, as we propose with SCHC, does not only improve anonymization and prevents ossification but also reduces packet sizes, which is very beneficial in the constrained IoT.

Obfuscating meta-data from DNS operators. Oblivious DNS [46], [86] provides means to decouple client information from DNS queries by encrypting query information and redirecting these queries to a special authoritative name server. Oblivious DoH [47], [48] extended this idea for DoH by employing a dedicated DoH proxy. Wang *et al.* [87] introduced PINOT as an early alternative to that, which obfuscates source information directly on a programmable network switch. Kulkarni *et al.* [49] presented a port of Oblivious DoH for DoQ. However, obfuscating the client from the recursive resolver is different challenge than this paper tries to solve.

Comparing CoAP to other protocols. We compared CoAP to Information-Centric Networking (ICN) with Named Data Networks (NDN) and MQTT and found NDN to be the best performing protocol [88]. This later resulted in guidance on how to deploy ICNs using CoAP [89], [90]. Nedergaard *et al.* [91] compared the DNS transfer protocols we use in this paper, CoAP, secured with DTLS or OSCORE, and HTTPS. They found that CoAP, in both its security variants, has a performance advantage over HTTPS, even when using UDP-based HTTP/3 (over QUIC). We compared the performance of DoC to classic DNS over UDP and DNS over DTLS [17]. In this comparison, we included both security variants of CoAP. We found that DNS over OSCORE outperforms DNS over CoAPS and DNS over DTLS.

Classifying encrypted DNS traffic and mitigation. Wickramasinghe *et al.* [92] found that most datasets used in studies that classify network traffic do not reflect current protocols (specifically TLS v1.3) and that most state-of-the-art classifiers are prone to overfitting. They provided best practices to mitigate these issues and a new dataset. Houser *et al.* [32] investigated the classification of DNS over TLS (DoT) traffic, including details of the queried webpage, by Random Forest (RF). Csikor *et al.* [33] performed one of the earliest evaluations of distinguishing DoH traffic from web traffic. They found that padding [39] does not suffice to blend DoH traffic into web traffic, and

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

provide extended padding techniques for DoH based on prior web traffic. Pélissier *et al.* [34] demonstrated that even with DoH, individual IoT devices can be correctly identified with high accuracy. Moulya and Hedge [36] introduced a fast and lightweight classification method using RF, Catboost, and XGboost classifiers to rapidly distinguish DoH from data traffic as well as detection of malicious traffic within the DoH traffic. They found that feature vectors based on the packet length yield the best results. Typically, mitigations are based on EDNS(0) padding [38] or random delays. These are typically not compatible with the constrained IoT. Alyami *et al.* [67] proposed a randomized segmentation of TCP traffic as an alternative to large padding size. However, this is not possible with CoAP block wise transfer as all blocks needs to be of the same size.

Perdisci *et al.* [5] showed that their fingerprinting technique that is based on the domain names they query can identify IoT devices. Building on that we identified IoT devices based on the domain names using more common classification techniques, such as Logistic Regression (LR), Support Vector Machine (SVM), or Random Forest (RF) [68]. In contrast to this study, both works only considered unencrypted messages and domain names.

8. Discussion of Limitations and Results

In this section, we discuss potential limitations of our proposal and provide broader implications of our findings.

8.1. Limitations of Approach

Threat model. We assume an eavesdropper is connected to the wireless domain, as wireless networks are common in IoT scenarios. This assumption simplifies the attacker task because the attacker does not need direct infrastructure access. However, assuming advanced attackers capable of wiretapping or accessing infrastructure devices does not contradict our observations. The SCHC-based mitigation can be deployed on any network device between the gateway and proxy or gateway and server.

Method. We focused on analyzing packet structures and using classic ML-classifiers, as temporal obfuscation techniques proposed in prior research [32], [34], [36] are ineffective in IoT scenarios. This ineffectiveness arises because of two reasons. First, low-power wireless networks with slow and random medium access, which is orders of magnitudes slower than the larger Internet, cannot cope with additional artificial delays or other common temporal obfuscation techniques [34]. Second, system behaviors of IoT devices require fast DNS resolution. After boot-up or long sleep cycles, *i.e.,* when cached results are likely expired, data must be retrieved before the device goes back to sleep. Therefore, our proposal provides obfuscation techniques orthogonal to temporal features.

Furthermore, we used established techniques to ensure comparability with prior work. Deep-learning techniques will likely yield higher accuracy when distinguishing DNS traffic, but conflict with this objective.

Dataset, including usage of HTTP Archive. To the best of our knowledge, there is currently no freely available, large-scale dataset for constrained IoT traffic that includes both structured data objects and resource locators (URLs) referencing these objects. When collecting data from the HTTP Archive [31], we aimed for a tradeoff between the availability of Internet-based M2M communication data (JSON objects) and a reasonable alignment to the constrained IoT. While the HTTP Archive only contains Web requests and responses, these also include many JSON-based REST API calls. Even though not a perfect match, they fit the M2M communication patterns of CoAP when accounting for the smaller IoT packet sizes. While largely synthesized from real-world data, we believe our data corpus can serve as valuable input for future studies on DNS privacy. Features such as packet lengths can be leveraged in unconstrained scenarios, while the SCHC and block-wise parts offer insights into compressed and fragmented IoT traffic, in which these features are omitted or adjusted. The Onion OSCORE part offers insights into IoT traffic, which either obfuscates or randomizes many features. We note the caveat that URI paths of data objects are longer than in typical CoAP scenarios since they are inherited from HTTP traffic. We do not recommend this dataset for isolated DoH studies, though, because we intentionally weakened encryption using TLS 1.2 for better comparison of HTTPS and CoAP.

8.2. Discussion of Results

How should our results be interpreted? Our techniques do not minimize distinction accuracy, nor was this our objective. Our aim was to demonstrate and quantify the (notable) effect of common standard IoT protocols on lowering the detection accuracy. Combining with other, already established obfuscation techniques may help to reduce distinction further. The lowest achievable accuracy is hard to determine and depends on many factors, such as individual costs or gain [93]. 50% accuracy, which represent an 'attack' of randomly guessing the type of packet, is typically understood as the lowest accuracy necessary to render an attack useless [94]. Current results are much larger than that. Our results for DoH also show higher accuracy than prior work. We did not analyze this in detail but it may easily be explained by (*i*) we use weaker encryption with TLS-PSK and TLS v1.2 to stay comparable to CoAPS (see Section 3.3) and (*ii*) we look at the full packet as feature vector, instead of more scalar features, providing more potential for ML-classifiers to distinguish packets (see Section 5.1) in exchange for slower computation (Section 5.3 shows training times of mean 10 minutes per dataset).

Can SCHC header compression and CoAP block-wise transfer decrease distinction of DNS from data traffic? Yes. Replacing information with an opaque rule ID using peer-based rule sets decreases the capability of a classifier to distinguish encrypted DNS and data traffic. In SCHC scenarios, we observe lower accuracy than in unconstrained scenarios throughout our header field analysis. However, in block-wise transfer scenarios, the last block often leaks enough information to keep the accuracy above 50%. This could be circumvented by using the padding option proposed in [95] to extend only the last block to full block length. Compared to using EDNS(0) padding, this has the additional advantage that data packets can be padded, which improves obfuscation. As the block size is a fixed known length, prior knowledge of data traffic,

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

e.g., as proposed by Csikor *et al.* [33], is not required for a padding strategy. This reduces complexity, which is particularly important for constrained devices. Like EDNS(0) padding, our solutions are based on pre-established or proposed IETF standards [23]–[27], [95] and merely require reconfiguration, or in case of the CoAP padding option, small updates. SCHC rules, including peer identities, can either be directly shipped with the firmware updates or updated via a YANG data model [23]. With CORECONF [24]–[26], we can even reuse the same CoAP-based obfuscation techniques (*i.e.,* encrypted block-wise transfer). Placing the SCHC compressor/decompressor closer to the proxy or servers extends our SCHC solution outside the wireless domain. If two SCHC hops know about each other, *e.g.,* using CORECONF or more generally NETCONF [96], our proposal can also be deployed beyond the constrained domain.

Should header data be obscured? With the introduction of QUIC and Encrypted Client Hello in TLS, the practicality of obfuscating transport header information has been widely discussed. In constrained IoT scenarios, the main concerns relate (*i*) to message sizes, (*ii*) ossifying fields to enable compression, or (*iii*) endpoints being unaware of the constrained link [84]. We argue for not sending certain network, transport, or application header fields at all. Since we propose the usage of SCHC, a technique established for constrained networks, a negative effect on message size and compression is not expected. SCHC provides fragmentation if needed, and compression can be used for obfuscation.

How does `application/dns+cbor` encoding impact the distinction of DNS? Throughout our evaluation, using `application/dns+cbor` yields either comparable or better results than using the classic DNS message format, especially when using CBOR as a data format because it aligns lengths. As such, depending on the use case and network setup, we recommend `application/dns+cbor` to make DNS and data traffic harder to distinguish. More improvement is provided by CoAP block-wise transfer (especially if the block option is protected by CoAPS or OSCORE), which aligns the lengths of frames to a fixed size.

Which obfuscation techniques should I use? Encryption, equaling out packet lengths, and header elision together provide a solid foundation for obfuscation. Peer-based SCHC rules reduces the need for a centralized proxy to increase obfuscation of traffic patterns when link layer addresses are randomized. These peers do not need to be multiple servers. Sharing the load over *multiple* proxies is also viable, *e.g.,* to use caching advantages of proxies. When using peer-based SCHC rules, the overhead of Onion OSCORE also becomes unnecessary, as relevant OSCORE information can be obfuscated by header elision. An extension of DoC for Oblivious DNS [46] to anonymize clients from recursive resolvers would help to reduce the risk of Internet centralization even further.

Does DoC provide better privacy than DoH? With RF we observe lower accuracy for DoH than for DoC. Our header field analysis points to the additional affirming counter in TinyDTLS, which uses the record epoch and sequence number as explicit nonce. Apart from this specific implementation behavior, DoC over OSCORE together with SCHC can obfuscate DNS traffic in data traffic even better than our weakened DoH setup can. We found that randomized counters, in case of CoAP MID and using tokens, and the obfuscation of OSCORE Partial IV, as proposed in [73], can significantly contribute to this advantage. These mechanisms cannot easily be copied to DoH when using HTTP/2, because order-preserving sequence numbers in TCP cannot be randomized or obfuscated. In DoH, HTTP/3 would be needed where the packet numbers of QUIC and their lengths are also protected by encryption [97], [98].

9. Conclusion

This paper closes a gap by analyzing DNS over CoAP (DoC) as an obfuscation technique for DNS. Prior work only investigated DNS over TLS (DoT), DNS over HTTPS (DoH), or DNS over QUIC (DoQ). We provide a data corpus that provides potential insights for future studies into DoC. Using this data corpus we utilize permutation importance for Random Forest to investigate which header leaks information. We show that DNS over CoAP (DoC) can provide good obfuscation and provide guidelines how to improve it even further. These include using Object Security for Constrained RESTful Environments (OSCORE) [29] with randomized identifiers such as message IDs and tokens and block-wise transfer. We can even improve on that by using the constrained use case as an advantage and omit leaking information altogether using Static Context Header Compression (SCHC). When using dedicated header compression rule sets for the upstream DNS and data servers we get their accuracy down to 77%, similar to a proxy-based approach, closing identified leaks.

Our data corpus has potential for extension which will inform future work. Once DTLS 1.3 implementations reach suitable stability with CoAP, the data corpus should be amended with such traces. Implementation of encrypted OSCORE Partial IV [73] or the CoAP padding option [95] will also provide material for amendment. The advent of Oblivious DNS provides potential of an analysis how an oblivious version of DoC performs. Real world traces of DoC, especially when mixing with or even being used for classic Internet deployments, will provide valuable insights into the privacy enhancements of DoC once it sees wider use.

Artifacts. All artifacts of this work, including the data corpus of traffic traces, implementations, and experiment results, as well the complete plots of our header field analysis are available under **doi:10.5281/zenodo.19697848**.

Acknowledgment

We would like to thank the anonymous shepherd and reviewers for their valuable feedback on this paper. Special thanks to Christian Amsüss for his help and guidance in beating *aiocoap* into shape for the evaluation of this paper and Mikolai Gütschow for reviewing the artifacts. This work was supported in parts by the German Federal Ministry of Research, Technology and Space (BMFTR) within the research project C-ray4edge (grants 16KIS1694K and 16KIS1695).

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

References

[1] P. Mockapetris, "Domain names - implementation and specification," IETF, RFC 1035, November 1987. [Online]. Available: https://doi.org/10.17487/RFC1035

[2] L. Zhu, Z. Hu, J. Heidemann, D. Wessels, A. Mankin, and N. Somaiya, "Connection-Oriented DNS to Improve Privacy and Security," in *Proc. of IEEE Symposium on Security and Privacy*. Piscataway, NJ, USA: IEEE, May 2015, pp. 171–186. [Online]. Available: https://doi.org/10.1109/SP.2015.18

[3] D. Wook Kim and J. Zhang, "Deriving and measuring DNS-based fingerprints," *Journal of Information Security and Applications*, vol. 36, pp. 32–42, 2017. [Online]. Available: https://doi.org/10.1016/j.jisa.2017.07.006

[4] N. Apthorpe, D. Y. Huang, D. Reisman, A. Narayanan, and N. Feamster, "Keeping the Smart Home Private with Smart(er) IoT Traffic Shaping," in *Proc. on Privacy Enhancing Technologies Symposium*, vol. 2019, 2019, pp. 128–148. [Online]. Available: https://doi.org/10.2478/popets-2019-0040

[5] R. Perdisci, T. Papastergiou, O. Alrawi, and M. Antonakakis, "IoTFinder: Efficient Large-Scale Identification of IoT Devices via Passive DNS Traffic Analysis," in *IEEE EuroS&P 2020*, 2020, pp. 474–489. [Online]. Available: https://doi.org/10.1109/EuroSP48549.2020.00037

[6] D. Chang, J. Q. Chen, Z. Li, and X. Li, "Hide and Seek: Revisiting DNS-based User Tracking," in *Proc. of IEEE Euro S&P*. Piscataway, NJ, USA: IEEE, 2022, pp. 188–205. [Online]. Available: https://10.1109/EuroSP53844.2022.00020

[7] C. Rossow, "Amplification Hell: Revisiting Network Protocols for DDoS Abuse," in *Proc. of NDSS*. Internet Society, 2014. [Online]. Available: https://doi.org/10.14722/ndss.2014.23233

[8] D. C. MacFarland, C. A. Shue, and A. J. Kalafut, "The Best Bang for the Byte: Characterizing the Potential of DNS Amplification Attacks," *Computer Networks*, vol. 116, pp. 12–21, 2017. [Online]. Available: https://doi.org/10.1016/j.comnet.2017.02.007

[9] H. Cristian, K. Merike, C. Lyman, C. Kimberly, S. Mark, M. Danny *et al.*, "The DNS in IoT: Opportunities, Risks, and Challenges," *IEEE Internet Computing*, vol. 24, no. 4, pp. 23–32, 2020. [Online]. Available: https://doi.org/10.1109/MIC.2020.3005388

[10] M. Nawrocki, M. Jonker, T. C. Schmidt, and M. Wählisch, "The Far Side of DNS Amplification: Tracing the DDoS Attack Ecosystem from the Internet Core," in *Proc. of ACM Internet Measurement Conference (IMC)*. New York: ACM, 2021, pp. 419–434. [Online]. Available: https://doi.org/10.1145/3487552.3487835

[11] S. Adiwal, B. Rajendran, S. D. Sudarsan *et al.*, "DNS Intrusion Detection (DID) — A Snort-based Solution to Detect DNS Amplification and DNS Tunneling Attacks," *Franklin Open*, vol. 2, pp. 1–11, 2023. [Online]. Available: https://doi.org/10.1016/j.fraope.2023.100010

[12] M. Koch, F. Dolzmann, T. C. Schmidt, and M. Wählisch, "Forward to Hell? On the Potentials of Misusing Transparent DNS Forwarders in Reflective Amplification Attacks," in *Proc. of ACM Conference on Computer and Communications Security (CCS)*. New York: ACM, November 2025, pp. 3915–3929. [Online]. Available: https://doi.org/10.1145/3719027.3765096

[13] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)," IETF, RFC 7858, May 2016. [Online]. Available: https://doi.org/10.17487/RFC7858

[14] P. Hoffman and P. McManus, "DNS Queries over HTTPS (DoH)," IETF, RFC 8484, October 2018. [Online]. Available: https://doi.org/10.17487/RFC8484

[15] C. Huitema, S. Dickinson, and A. Mankin, "DNS over Dedicated QUIC Connections," IETF, RFC 9250, May 2022. [Online]. Available: https://doi.org/10.17487/RFC9250

[16] M. S. Lenders, C. Amsüss, C. Gündoğan, T. C. Schmidt, and M. Wählisch, "DNS over CoAP (DoC)," IETF, RFC 9953, March 2026. [Online]. Available: https://doi.org/10.17487/RFC9953

[17] M. S. Lenders, C. Amsüss, C. Gündogan, M. Nawrocki, T. C. Schmidt, and M. Wählisch, "Securing Name Resolution in the IoT: DNS over CoAP," *Proceedings of the ACM on Networking (PACMNET)*, vol. 1, no. CoNEXT2, pp. 6:1–6:25, September 2023. [Online]. Available: https://doi.org/10.1145/3609423

[18] "DNS over TLS vs. DNS over HTTPS — Secure DNS," https://www.cloudflare.com/learning/dns/dns-over-tls/, [Online]. Accessed March 31, 2026.

[19] O. Thompson, A. M. Mandalari, and H. Haddadi, "Rapid IoT Device Identification at the Edge," in *Proceedings of the 2nd ACM International Workshop on Distributed Machine Learning*. New York, NY, USA: ACM, Dec. 2021, pp. 22—28. [Online]. Available: https://doi.org/10.1145/3488659.3493777

[20] H. Tschofenig, J. Arkko, D. Thaler, and D. McPherson, "Architectural Considerations in Smart Object Networking," IETF, RFC 7452, March 2015. [Online]. Available: https://doi.org/10.17487/RFC7452

[21] O. Garcia-Morchon, S. Kumar, and M. Sethi, "Internet of Things (IoT) Security: State of the Art and Challenges," IETF, RFC 8576, April 2019. [Online]. Available: https://doi.org/10.17487/RFC8576

[22] A. Minaburo, L. Toutain, C. Gomez, D. Barthel, and J. Zuniga, "SCHC: Generic Framework for Static Context Header Compression and Fragmentation," IETF, RFC 8724, April 2020. [Online]. Available: https://doi.org/10.17487/RFC8724

[23] A. Minaburo and L. Toutain, "A YANG Data Model for Static Context Header Compression (SCHC)," IETF, RFC 9363, March 2023. [Online]. Available: https://doi.org/10.17487/RFC9363

[24] A. Minaburo, L. Toutain, J. A. FERNANDEZ, C. Banier, and M. Dumay, "CORECONF Rule management for SCHC," IETF, Internet-Draft – work in progress 01, October 2025. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-toutain-schc-coreconf-management-01

[25] Q. Lampin, A. Minaburo, M. Tiloca, and L. Toutain, "Options representation in SCHC YANG Data Models," IETF, Internet-Draft – work in progress 01, October 2025. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-schc-universal-option-01

[26] M. Veillette, P. V. der Stok, A. Pelov, A. Bierman, and C. Bormann, "CoAP Management Interface (CORECONF)," IETF, Internet-Draft – work in progress 21, March 2026. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-core-comi-21

[27] C. Bormann and Z. Shelby, "Block-Wise Transfers in the Constrained Application Protocol (CoAP)," IETF, RFC 7959, August 2016. [Online]. Available: https://doi.org/10.17487/RFC7959

[28] M. S. Lenders, C. Bormann, M. Gütschow, T. C. Schmidt, and M. Wählisch, "A Concise Binary Object Representation (CBOR) of DNS Messages," IETF, Internet-Draft – work in progress 16, February 2026. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-lenders-dns-cbor-16

[29] G. Selander, J. Mattsson, F. Palombini, and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)," IETF, RFC 8613, July 2019. [Online]. Available: https://doi.org/10.17487/RFC8613

[30] C. Amsüss, M. Tiloca, and R. Höglund, "Using onion routing with CoAP," IETF, Internet-Draft – work in progress 04, July 2025. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-amsuess-t2trg-onion-coap-04

[31] HTTP Archive, "Getting started accessing the HTTP Archive with BigQuery," https://github.com/HTTPArchive/har.fyi/blob/a61dc0b/src/content/docs/guides/getting-started.mdx?plain=1, Nov. 2024, [Online]. Accessed March 31, 2026.

[32] R. Houser, Z. Li, C. Cotton, and H. Wang, "An Investigation on Information Leakage of DNS over TLS," in *Proc. of ACM CoNEXT*. New York, NY, USA: ACM, Dec. 2019, pp. 96–109. [Online]. Available: https://doi.org/10.1145/3359989.3365429

[33] L. Csikor, H. Singh, M. S. Kang, and D. M. Divakaran, "Privacy of DNS-over-HTTPS: Requiem for a Dream?" in *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, Sep. 2021, pp. 252–271. [Online]. Available: https://doi.org/10.1109/EuroSP51992.2021.00026

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

[34] S. Pélissier, G. Anselmi, A. K. Mishra, A. M. Mandalari, and M. Cunche, "Enhancing IoT Privacy: Why DNS-over-HTTPS Alone Falls Short?" in *2024 IEEE 23rd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. Piscataway, NJ, USA: IEEE, Dec. 2024, pp. 1353–1360. [Online]. Available: https://doi.org/10.1109/TrustCom63139.2024.00189

[35] G. Hu and K. Fukuda, "Privacy Leakage of DNS over QUIC: Analysis and Countermeasure," in *2024 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, Feb. 2024, pp. 518–523. [Online]. Available: https://doi.org/10.1109/ICAIIC60209.2024.10463369

[36] D. M. Moulya and S. Hegde, "Detection and Categorization of DNS over HTTPS Traffic Using Lightweight Feature Selection Methods and Ensemble Classification Model," in *Advanced Computing and Communications: Responsible AI*, S. Srinivasa, S. Saragur, and S. Malapaka, Eds. Springer Nature Switzerland, Jul. 2025, pp. 25–43. [Online]. Available: https://doi.org/10.1007/978-3-031-96473-2_3

[37] L. Csikor, Z. Lian, H. Zhang, N. Lakshmanan, and D. M. Divakaran, "DNS-over-QUIC and HTTP/3 in the Era of Transformers: The New Internet Privacy Battle," *IEEE Communications Magazine*, pp. 1–7, Jun. 2025. [Online]. Available: https://doi.org/10.1109/MCOM.004.2400680

[38] A. Mayrhofer, "The EDNS(0) Padding Option," IETF, RFC 7830, May 2016. [Online]. Available: https://doi.org/10.17487/RFC7830

[39] A. Mayrhofer, "Padding Policies for Extension Mechanisms for DNS (EDNS(0))," IETF, RFC 8467, October 2018. [Online]. Available: https://doi.org/10.17487/RFC8467

[40] M. S. Lenders, T. C. Schmidt, and M. Wählisch, "Fragment Forwarding in Lossy Networks," *IEEE Access*, vol. 9, pp. 143 969–143 987, October 2021. [Online]. Available: https://doi.org/10.1109/ACCESS.2021.3121557

[41] Z. Shelby, K. Hartke, and C. Bormann, "The Constrained Application Protocol (CoAP)," IETF, RFC 7252, June 2014. [Online]. Available: https://doi.org/10.17487/RFC7252

[42] E. Rescorla and N. Modadugu, "Datagram Transport Layer Security Version 1.2," IETF, RFC 6347, January 2012. [Online]. Available: https://doi.org/10.17487/RFC6347

[43] E. Rescorla, H. Tschofenig, and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3," IETF, RFC 9147, April 2022. [Online]. Available: https://doi.org/10.17487/RFC9147

[44] P. F. Syverson, M. G. Reed, and D. M. Goldschlag, "Private web browsing," *Journal of Computer Security*, vol. 5, no. 3, pp. 237–248, Jul. 1997. [Online]. Available: https://doi.org/10.3233/JCS-1997-5305

[45] R. Dingledine, N. Mathewson, P. F. Syverson *et al.*, "Tor: The Second-Generation Onion Router," in *Proc. of 2004 USENIX Security Symposium*, vol. 4. San Diego, CA, USA: USENIX Association, 2004, pp. 303–320. [Online]. Available: https://www.usenix.org/conference/13th-usenix-security-symposium/tor-second-generation-onion-router

[46] P. Schmitt, A. Edmundson, A. Mankin, and N. Feamster, "Oblivious DNS: Practical Privacy for DNS Queries," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, pp. 228–224, 2019. [Online]. Available: https://doi.org/10.2478/popets-2019-0028

[47] S. Singanamalla, S. Chunhapanya, J. Hoyland, M. Vavruša, T. Verma, P. Wu *et al.*, "Oblivious DNS over HTTPS (ODoH): A Practical Privacy Enhancement to DNS," *Proceedings on Privacy Enhancing Technologies*, vol. 2021, pp. 575–592, 2021. [Online]. Available: https://doi.org/10.2478/popets-2021-0085

[48] E. Kinnear, P. McManus, T. Pauly, T. Verma, and C. Wood, "Oblivious DNS over HTTPS," IETF, RFC 9230, June 2022. [Online]. Available: https://doi.org/10.17487/RFC9230

[49] A. Kulkarni, T. Das, and V. Balachandran, "ODoQ: Oblivious DNS-over-QUIC," in *2025 17th International Conference on COMmunication Systems and NETworks (COMSNETS)*, Jan. 2025, pp. 36–41. [Online]. Available: https://doi.org/10.1109/COMSNETS63942.2025.10885687

[50] S. Farrell, "Low-Power Wide Area Network (LPWAN) Overview," IETF, RFC 8376, May 2018. [Online]. Available: https://doi.org/10.17487/RFC8376

[51] C. Gomez and A. Minaburo, "Transmission of SCHC-compressed packets over IEEE 802.15.4 networks," IETF, Internet-Draft – work in progress 12, February 2026. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-6lo-schc-15dot4-12

[52] M. Blanchet, W. Eddy, and T. Li, "An Architecture for IP in Deep Space," IETF, Internet-Draft – work in progress 03, March 2026. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-many-tiptop-ip-architecture-03

[53] C. Bormann and P. Hoffman, "Concise Binary Object Representation (CBOR)," IETF, RFC 8949, December 2020. [Online]. Available: https://doi.org/10.17487/RFC8949

[54] S. Farrell and H. Tschofenig, "Pervasive Monitoring Is an Attack," IETF, RFC 7258, May 2014. [Online]. Available: https://doi.org/10.17487/RFC7258

[55] R. Moskowitz, P. Thubert, C. Gomez, A. Minaburo, and M. Blanchet, "Protocol Numbers for SCHC," IETF, Internet-Draft – work in progress 06, December 2025. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-schc-protocol-numbers-06

[56] curl project, "curl," https://curl.se/docs/manpage.html, May 2025, [Online]. Accessed March 31, 2026.

[57] M. Knodel, W. Hardaker, and T. Pauly, "Report from the IAB Workshop on Management Techniques in Encrypted Networks (M-TEN)," IETF, RFC 9490, January 2024. [Online]. Available: https://doi.org/10.17487/RFC9490

[58] C. Amsüss and M. S. Lenders, "CoAP Transport Indication," IETF, Internet-Draft – work in progress 09, July 2025. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-core-transport-indication-09

[59] A. Grönholm and K. Smallwood, "cbor2 5.6.2 – PyPI," https://pypi.org/project/cbor2/5.6.5/, Feb. 2024, [Online]. Accessed March 31, 2026.

[60] M. S. Lenders, "cbor4dns," https://github.com/netd-tud/cbor4dns/commit/0e5d81b5, Feb. 2025, [Online]. Accessed March 31, 2026.

[61] M. S. Lenders, C. Bormann, T. C. Schmidt, and M. Wählisch, "A Concise Binary Object Representation (CBOR) of DNS Messages," IETF, Internet-Draft – work in progress 10, November 2024. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-lenders-dns-cbor-10

[62] J. Hugon, M. Cunche, and T. Begin, "Roma: rotating mac address for privacy protection," in *Proceedings of the SIGCOMM '22 Poster and Demo Sessions*, ser. SIGCOMM '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 31–33. [Online]. Available: https://doi.org/10.1145/3546037.3546055

[63] A. Minaburo, L. Toutain, and R. Andreasen, "Static Context Header Compression (SCHC) for the Constrained Application Protocol (CoAP)," IETF, RFC 8824, June 2021. [Online]. Available: https://doi.org/10.17487/RFC8824

[64] M. Tiloca, L. Toutain, I. Martínez, and A. Minaburo, "Static Context Header Compression (SCHC) for the Constrained Application Protocol (CoAP)," IETF, Internet-Draft – work in progress 07, December 2025. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-schc-8824-update-07

[65] A. Fragkiadakis, "DTLS Static Context Header Compression - Implementation and Evaluation in the Contiki-NG," in *2022 IEEE Conference on Standards for Communications and Networking (CSCN)*. Piscataway, NJ, USA: IEEE, Nov. 2022, pp. 131–137. [Online]. Available: https://doi.org/10.1109/CSCN57023.2022.10051055

[66] A. Jain, K. Nandakumar, and A. Ross, "Score normalization in multimodal biometric systems," *Pattern Recognition*, vol. 38, no. 12, pp. 2270–2285, 2005. [Online]. Available: https://doi.org/10.1016/j.patcog.2005.01.012

[67] M. Alyami, A. Alghamdi, M. Alkhowaiter, C. Zou, and Y. Solihin, "Random Segmentation: New Traffic Obfuscation against Packet-Size-Based Side-Channel Attacks," Open Archive: arXiv.org, Technical Report arXiv:2309.05941, Sep. 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2309.05941

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

[68] I. Ayoub, M. S. Lenders, B. Ampeau, S. Balakrichenan, K. Khawam, T. C. Schmidt *et al.*, "Toward a Better Understanding of IoT Domain Names: A Study of IoT Backend," *IEEE Access*, vol. 13, pp. 68 871–68 890, April 2025. [Online]. Available: https://doi.org/10.1109/ACCESS.2025.3561521

[69] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011. [Online]. Available: http://jmlr.org/papers/v12/pedregosa11a.html

[70] S. Raschka, J. Patterson, and C. Nolet, "Machine Learning in Python: Main developments and technology trends in data science, machine learning, and artificial intelligence," Open Archive: arXiv.org, Technical Report arXiv:2002.04803, Feb. 2020. [Online]. Available: https://arxiv.org/abs/2002.04803

[71] scikit-learn developers, "Permutation feature importances," https://scikit-learn.org/1.7/modules/permutation_importance.html, Jan. 2025, [Online]. Accessed March 31, 2026.

[72] A. K. Mishra, A. Losty, A. M. Mandalari, J. Mozley, and M. Cunche, "IoT DNS Security and Privacy Guidelines," IETF, Internet-Draft – work in progress 02, February 2026. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-iotops-iot-dns-guidelines-02

[73] M. Tiloca, J. P. Mattsson, R. Höglund, and G. Selander, "Stand-in Key Identifier and Encrypted Partial IV in the Constrained Application Protocol (CoAP) OSCORE Option," IETF, Internet-Draft – work in progress 02, March 2026. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-tiloca-core-oscore-piv-enc-02

[74] A. Hubert and R. van Mook, "Measures for Making DNS More Resilient against Forged Answers," IETF, RFC 5452, January 2009. [Online]. Available: https://doi.org/10.17487/RFC5452

[75] S. Bortzmeyer, "DNS Privacy Considerations," IETF, RFC 7626, August 2015. [Online]. Available: https://doi.org/10.17487/RFC7626

[76] H. Tschofenig, "[Iotops] Encrypted DNS and draft-mishra-iotops-iot-dns-guidelines," https://mailarchive.ietf.org/arch/msg/iotops/yxkMrOXsPYj0f21st8D3GNbVkSM/, Jul. 2025, mailing List Archive. [Online]. Accessed March 31, 2026.

[77] A. Losty, A. K. Mishra, M. Cunche, and A. M. Mandalari, "Towards Operational and Security Best Practices for DNS in the Internet of Things," in *Proc. of ANRW*. New York, NY, USA: ACM, 2025, p. 160–163. [Online]. Available: https://doi.org/10.1145/3744200.3744764

[78] E. Rescorla, K. Oku, N. Sullivan, and C. A. Wood, "TLS Encrypted Client Hello," IETF, RFC 9849, March 2026. [Online]. Available: https://doi.org/10.17487/RFC9849

[79] V. Shmatikov, "Anonymity is not privacy: Technical perspective," *Commun. ACM*, vol. 54, no. 12, p. 132, Dec. 2011. [Online]. Available: https://doi.org/10.1145/2043174.2043198

[80] H.-C. Hsiao, T. H.-J. Kim, A. Perrig, A. Yamada, S. C. Nelson, M. Gruteser *et al.*, "LAP: Lightweight Anonymity and Privacy," in *2012 IEEE Symposium on Security and Privacy*, 2012, pp. 506–520. [Online]. Available: https://doi.org/10.1109/SP.2012.37

[81] C. Chen, D. E. Asoni, D. Barrera, G. Danezis, and A. Perrig, "HORNET: High-speed Onion Routing at the Network Layer," in *Proc. of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: ACM, 2015, pp. 1441–1454. [Online]. Available: https://doi.org/10.1145/2810103.2813628

[82] D. L. Chaum, "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–90, Feb. 1981. [Online]. Available: https://doi.org/10.1145/358549.358563

[83] M. K. Reiter and A. D. Rubin, "Anonymous Web Transactions with Crowds," *Commun. ACM*, vol. 42, no. 2, pp. 32–48, Feb. 1999. [Online]. Available: https://doi.org/10.1145/293411.293778

[84] G. Fairhurst and C. Perkins, "Considerations around Transport Header Confidentiality, Network Operations, and the Evolution of Internet Transport Protocols," IETF, RFC 9065, July 2021. [Online]. Available: https://doi.org/10.17487/RFC9065

[85] D. Benjamin, "Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility," IETF, RFC 8701, January 2020. [Online]. Available: https://doi.org/10.17487/RFC8701

[86] A. Edmundson, P. Schmitt, N. Feamster, and A. Mankin, "Oblivious DNS - Strong Privacy for DNS Queries," IETF, Internet-Draft – work in progress 00, July 2018. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-annee-dprive-oblivious-dns-00

[87] L. Wang, H. Kim, P. Mittal, and J. Rexford, "Programmable in-Network Obfuscation of DNS Traffic," in *NDSS: DNS Privacy Workshop*, Feb. 2021. [Online]. Available: https://www.ndss-symposium.org/ndss-paper/auto-draft-129/

[88] C. Gündogan, P. Kietzmann, M. Lenders, H. Petersen, T. C. Schmidt, and M. Wählisch, "NDN, CoAP, and MQTT: A Comparative Measurement Study in the IoT," in *Proc. of 5th ACM Conference on Information-Centric Networking (ICN)*. New York, NY, USA: ACM, September 2018, pp. 159–171. [Online]. Available: https://doi.org/10.1145/3267955.3267967

[89] C. Gündogan, C. Amsüss, T. C. Schmidt, and M. Wählisch, "IoT Content Object Security with OSCORE and NDN: A First Experimental Comparison," in *Proc. of 19th IFIP Networking Conference*. Piscataway, NJ, USA: IEEE Press, June 2020, pp. 19–27. [Online]. Available: https://ieeexplore.ieee.org/document/9142731

[90] C. Gündogan, C. Amsüss, T. C. Schmidt, and M. Wählisch, "Content Object Security in the Internet of Things: Challenges, Prospects, and Emerging Solutions," *IEEE Transactions on Network and Service Management (TNSM)*, vol. 19, no. 1, pp. 538–553, March 2022. [Online]. Available: https://doi.org/10.1109/TNSM.2021.3099902

[91] K. Nedergaard, B. Singh, and B. Andersen, "Evaluating CoAP, OSCORE, DTLS and HTTPS for Secure Device Communication," in *Internet of Everything*, T. Pereira, J. Impagliazzo, and H. Santos, Eds. Springer Nature Switzerland, 2023, pp. 117–132. [Online]. Available: https://doi.org/10.1007/978-3-031-25222-8_10

[92] N. Wickramasinghe, A. Shaghaghi, G. Tsudik, and S. Jha, "SoK: Decoding the Enigma of Encrypted Network Traffic Classifiers," in *2025 IEEE Symposium on Security and Privacy (SP)*. Piscataway, NJ, USA: IEEE, May 2025, pp. 1825–1843. [Online]. Available: https://doi.org/10.1109/SP61157.2025.00165

[93] D. Bodeau and R. Graubart, "Characterizing Effects on the Cyber Adversary: A Vocabulary for Analysis and Assessment," The MITRE Corporation, MITRE Technical Report MTR130432, Nov. 2013. [Online]. Available: https://apps.dtic.mil/sti/pdfs/AD1107912.pdf

[94] J. Li, N. Li, and B. Ribeiro, "Membership Inference Attacks and Defenses in Classification Models," in *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy (CODASPY)*. New York, NY, USA: ACM, Apr. 2021, pp. 5—16. [Online]. Available: https://doi.org/10.1145/3422337.3447836

[95] C. Amsüss and M. Tiloca, "End-to-End Protected and Cacheable Responses for the Constrained Application Protocol (CoAP) using Group Object Security for Constrained RESTful Environments (Group OSCORE)," IETF, Internet-Draft – work in progress 01, March 2026. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-core-cacheable-oscore-01

[96] R. Enns, "NETCONF Configuration Protocol," IETF, RFC 4741, December 2006. [Online]. Available: https://doi.org/10.17487/RFC4741

[97] J. Iyengar and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport," IETF, RFC 9000, May 2021. [Online]. Available: https://doi.org/10.17487/RFC9000

[98] M. Thomson and S. Turner, "Using TLS to Secure QUIC," IETF, RFC 9001, May 2021. [Online]. Available: https://doi.org/10.17487/RFC9001

Appendix A.
Random Forest Results for All Scenarios

Figures 17 to 20 show the mean accuracy, precision, recall, and F_1 score for 5-fold cross validation of Random

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

Forest as annotated heatmaps. The x-axes summarize the link layer mode (unconstrained or SCHC and which rule set was used with SCHC) and the block size. The y-axes summarize network architecture (D1, P1, P2, D2), protocol (HTTPS, CoAPS, Onion OSCORE, CoAP) and format combination (JSON & CBOR, CBOR & CBOR, CBOR & DNS, JSON & DNS), roughly ordered by metric and group.

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

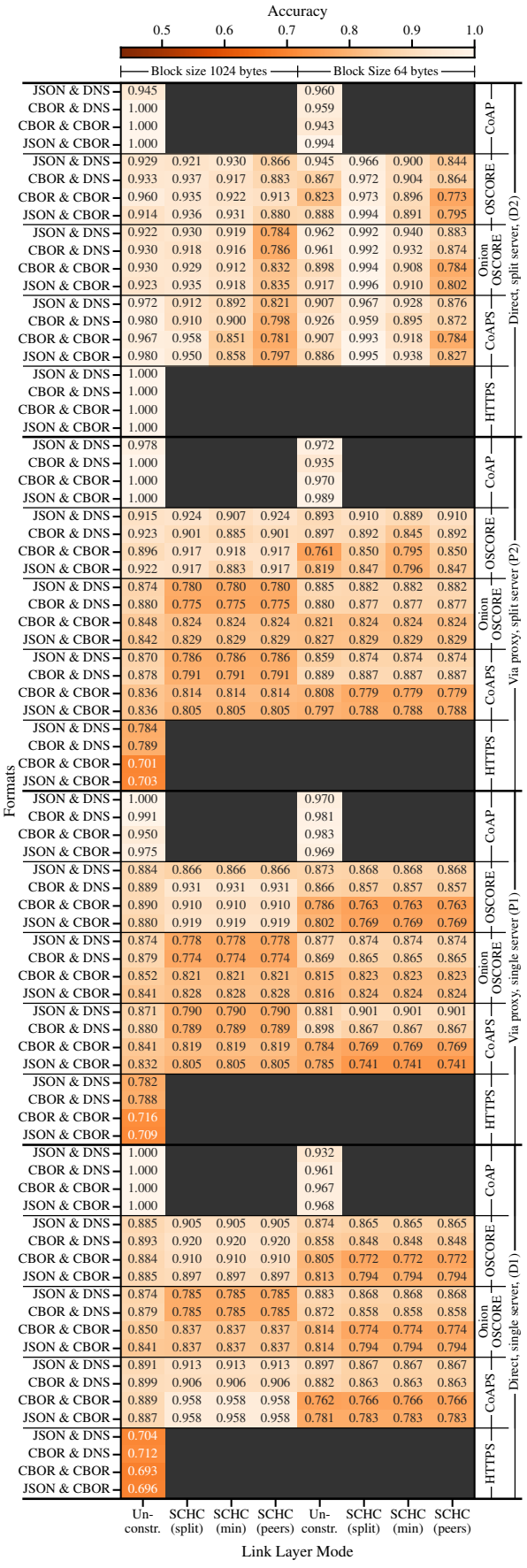


Figure 17: Mean accuracy for 5-fold cross validation of Random Forest. Dark gray cells mean that the scenario was not tested by design.

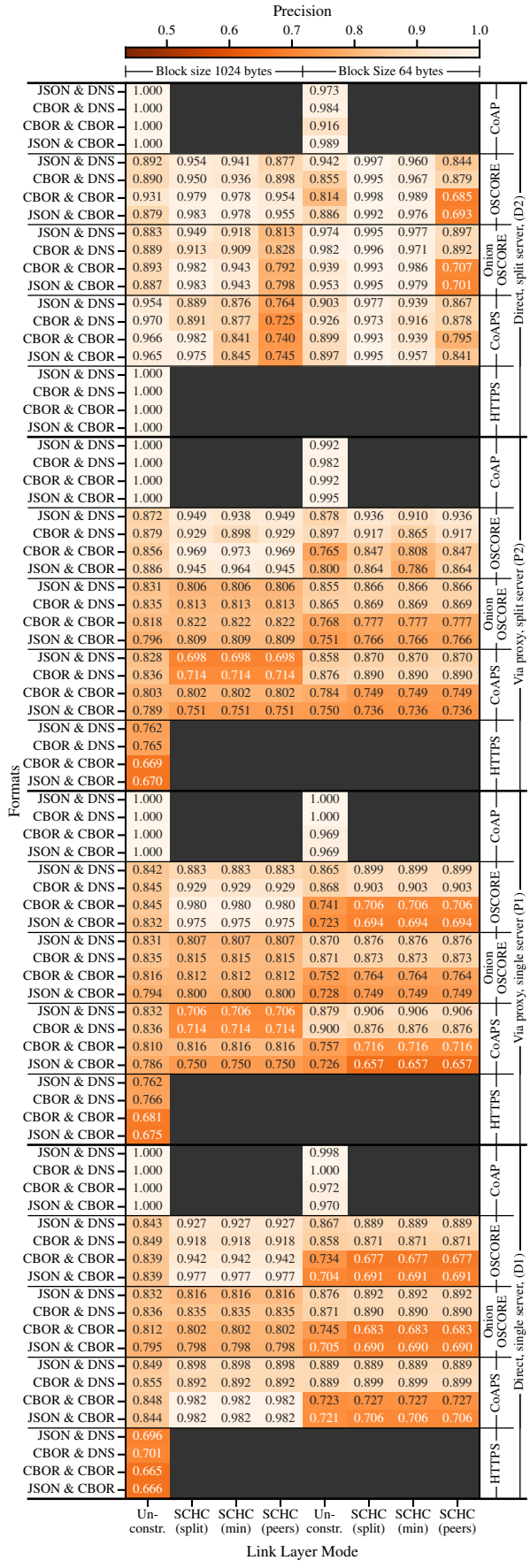


Figure 18: Mean precision for 5-fold cross validation of Random Forest. Dark gray cells mean that the scenario was not tested by design.

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.

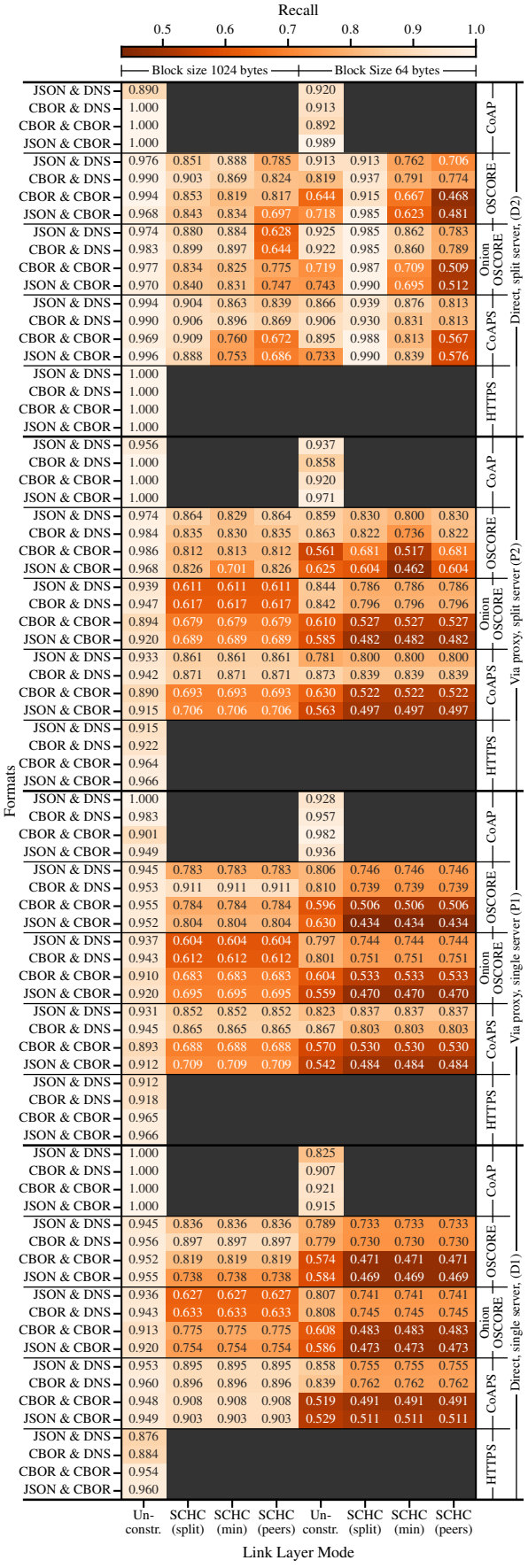


Figure 19: Mean recall for 5-fold cross validation of Random Forest. Dark gray cells mean that the scenario was not tested by design.

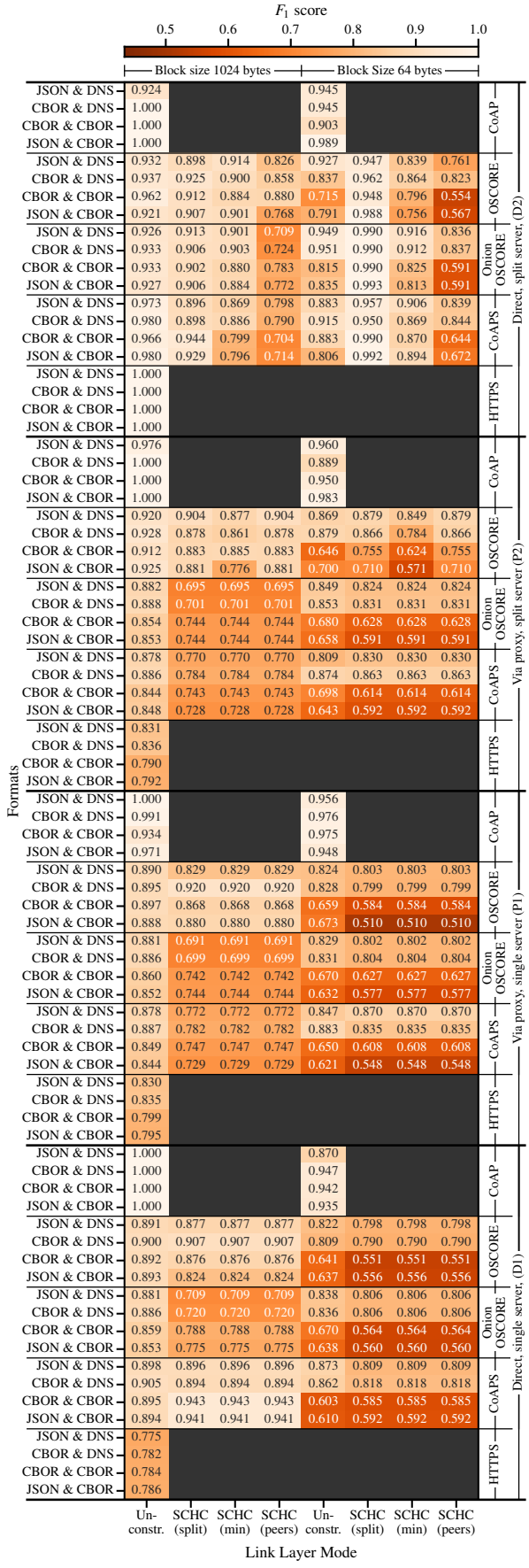


Figure 20: Mean F_1 score for 5-fold cross validation of Random Forest. Dark gray cells mean that the scenario was not tested by design.

Authorized licensed use limited to: SLUB Dresden. Downloaded on August 08,2026 at 08:14:43 UTC from IEEE Xplore. Restrictions apply.