Intermediate Certificate Suppression in Post-Quantum TLS: An Approximate Membership Querying Approach*

Dimitrios Sikeridis† Sean Huntley‡ David Ott§ Michael Devetsikiotis¶

December 8, 2023

Abstract

Quantum computing advances threaten the security of today's public key infrastructure, and have led to the pending standardization of alternative, quantum-resistant key encapsulation and digital signature cryptography schemes. Unfortunately, authentication algorithms based on the new post-quantum (PQ) cryptography create significant performance bottlenecks for TLS due to larger certificate chains which introduce additional packets and round-trips. The TLS handshake slowdown will be unacceptable to many applications, and detrimental to the broader adoption of quantum safe cryptography standards. In this paper, we propose a novel framework for Intermediate Certificate Authority (ICA) certificate suppression in TLS that reduces the authentication message size and prevents excessive round-trip delays. Our approach utilizes an approximate membership query (AMQ) data structure (probabilistic filter) to advertise known ICA certs to remote TLS endpoints so that unnecessary ICA certificates are omitted from the TLS handshake exchange. We showcase the extend of the PQ authentication overhead challenge in TLS, and evaluate the feasibility of AMQ filters for ICA suppression in terms of space and computational overhead. Finally, we experimentally evaluate the potential gains form our approach and showcase a 70% reduction in exchanged ICA cert data that translates to 15-50 MB of savings in PQ TLS and for certain Web-based application scenarios.

1 Introduction

Transport Layer Security (TLS) is a well-known and widely adopted secure communications protocol for establishing encrypted tunnels for mobile apps [43], accessing email servers [19], and web page transfer [10,26,34]). Studies suggest that over 60% of Internet connections [10,34], and ≈95% of Google services' connections [18] are implemented over the TLS-based secure HTTPS. In 2018, the latest version —TLS 1.3— was published as RFC 8446 [44], following an unprecedented standardization process. Since then, TLS 1.3 has been under deployment faster than any previous security protocol, and is currently used in ≈ 20% of observed connections, and ≈ 30% in the popular domain space [20].

*Accepted at the 18th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '22), December 6–9, 2022, Roma, Italy, https://doi.org/10.1145/3555050.3569127

†VMware xLabs, email: sikeridisd@vmware.com

‡VMware xLabs, email: shuntley@vmware.com

§VMware Research, email: dott@vmware.com

¶Department of Electrical and Computer Engineering, The University of New Mexico, email: mdevets@unm.edu

Meanwhile, quantum computing has continued to advance, posing a well-publicized future threat to public key cryptography standards (RSA, ECDH, and ECDSA [2, 3, 32])*. This has led to a major initiative by the US National Institute of Standards and Technology (NIST) to standardize quantum-secure cryptography alternatives for key exchange and digital signatures [35]. And while the timeline for the existence of a large-scale quantum computer is uncertain, the possibility of the "store now, decrypt later" attack†, gives a sense of urgency to the transition to these post-quantum (PQ) schemes [36, 37]. Such a transition, will be proven a major challenge for TLS 1.3, especially since PQ signature schemes are considered major performance bottleneck in the TLS 1.3 handshake [55]. In July 2022, NIST concluded the 3rd selection round of PQ signature schemes, announcing Dilithium [14], Falcon [17], and SPHINCS$^+$ [5] as the algorithms for future standardization. Initial measurements indicate that, across all choices and different parameter sets, only the lowest security levels of Dilithium and Falcon have acceptable delay ranges in TLS 1.3 ($\approx$ 9-15 ms median) [55]. The remaining algorithms and configurations will increase TLS handshake time by 50 to 500 ms *at a minimum* which will be *unacceptable* to many TLS-reliant applications and broadly detrimental PQ adoption.

The impact of PQ-based authentication on the TLS 1.3 handshake can be attributed to larger signature sizes [55]. During the TLS 1.3 handshake, the entity under authentication (e.g., a client application) must transmit a certificate chain to the remote party to allow them to verify the valid certificate links leading to a trusted root Certificate Authority (CA). Since PQ certificates can be significantly larger (e.g., while three ECDSA 384 certs are $\approx$ 2.14 KB, three Rainbow Ia certs amount to $\approx$ 175.35 KB, see [55]), transmitted messages enabling this verification can exceed TCP's congestion window size and add unwanted round-trip packet exchanges to the handshake. Thus, *PQ certificate transmission becomes the dominant latency-adding factor in PQ TLS.* This observation is further reinforced in [54] where AVX2-optimized versions of the PQ schemes were tested in TLS and SSH [25] confirming that certificate exchange is the main performance issue, while lattice-based key exchange does not add significant load. Additionally, Cloudflare [60] demonstrated that 9-10 KB certificate chains cause double-digit PQ TLS handshake slowdown, and most importantly a number of middleboxes and clients cannot handle certificate chains that exceed the 10KB mark.

Contributions. In this paper, we address this problem by designing and optimizing a novel ICA certificate suppression mechanism that relies on approximate membership query filters. The frameworks can aid the migration of TLS 1.3 to PQ authentication schemes and even make current (i.e., conventional-scheme) handshakes faster. Our contributions are summarized as follows:

- We quantify the PQ authentication impact on TLS 1.3 handshakes.
- We propose a probabilistic filter-based mechanism that advertises known ICA certificates to peers and reduces excessive authentication data transmissions in PQ TLS.
- We evaluate the volume of intermediate certs in the wild and investigate the performance of our approach in terms of filter-space overhead, computational cost, filter high-load factor, and estimated PQ TLS handshake speed-up.
- Finally, we discuss security-related implications of our design, and possible solutions.

*Shor's quantum algorithm [42, 53]. Assuming a practical quantum computer (QC) was available, it would solve ECDL and IF problems in polynomial time which would render ECDSA, ECDH and RSA insecure.

†where encypted information assets are recorded now to be decrypted by quantum-computer-capable adversaries later

Structure. Section 2 reviews related solutions on improving PQ TLS performance, and Section 3 discusses the overhead of PQ authentication on TLS. Section 4 introduces our filter-based intermediate certificate suppression mechanism, and Section 5 evaluates its feasibility and performance. Finally, Section 6 reports security-related considerations, while Section 7 concludes this paper.

2 Related Work

A number of recent papers discuss ways to reduce TLS handshake time specifically when PQ authentication is used. The authors in [55] and [41] propose the use of different PQ signature algorithms within the same certificate chain. While this method reduces the authentication data exchanged during the handshake, it assumes that all peers support multiple signature algorithms, which is questionable given the current scheme adoption speed of the industry. An alternative solution is suggested in [54] via increasing the TCP's initial congestion window (`initcwnd`) to reduce the number of round-trips occurring within the handshake. However, as authors note, increasing the `initcwnd` abruptly can affect the loss rate in low bandwidth or congested connections.

The omission of the Intermediate CA (ICA) certificates during the handshake would presumably reduce the message size and prevent any round trip triggering. Since the industry constantly strives for faster handshakes, this idea of ICA certificate suppression has surfaced in the form of IETF RFC drafts. The work in [57] proposes a new TLS extension to inform the server that the client does not need the ICA certificates. However, the specifics of how the client decides whether or not to request the ICA certificates are not introduced. The Compact TLS (cTLS) IETF draft [45, § 5.1.3] proposes a similar method of omitting certificates by using a pre-established certificate dictionary. In this case, however, the client and server need to have agreed on a list of certs for the dictionary, which has also to be frequently synchronized and updated in case of expired or revoked certs. This would require a separate dedicated synchronization mechanism.

Similar to our approach, Kampanakis and Kallitsis in [23] make a case for omitting the intermediate CA certificates during the handshake in (D)TLS and QUIC towards a faster completion time after considering the size of PQ certificates. Their work also analyses the impact of PQ algorithms on QUIC's amplification protection mechanism. Their ICA suppression approach introduces caching mechanisms that allow each handshake party to request intermediate CA omission via a specific flag in the TLS initial client message. This proposal, however, requires from the client to retain a specific mapping between ICA certs and the respective server/peer.

Mozilla has launched an Intermediate CA Preloading framework since Firefox 75 (April '20) that loads a small ICA certificate set in the user's profile ($\approx$ 100 certs) [61]. While their target is to eliminate SEC_ERROR_UNKNOWN_ ISSUER errors caused when servers are configured incorrectly, it is a first step towards ICA certificate suppression in TLS.

Finally, Schwabe et al. introduced KEMTLS [51], a framework that utilizes PQ key encapsulation mechanisms (KEMs) in the leaf certificate. This allows the resulting certificate chain to be smaller by a few KBs compared to certificates that utilize lattice-based PQ schemes for signing.

3 PQ TLS Authentication Overhead

Quantum safety through new NIST PQ algorithms and standards comes at a price in the form of larger key and signature sizes. For example, ECDSA and RSA, which are widely deployed in industry authentication standards, utilize keys and signatures between 32 and 256 bytes. PQ

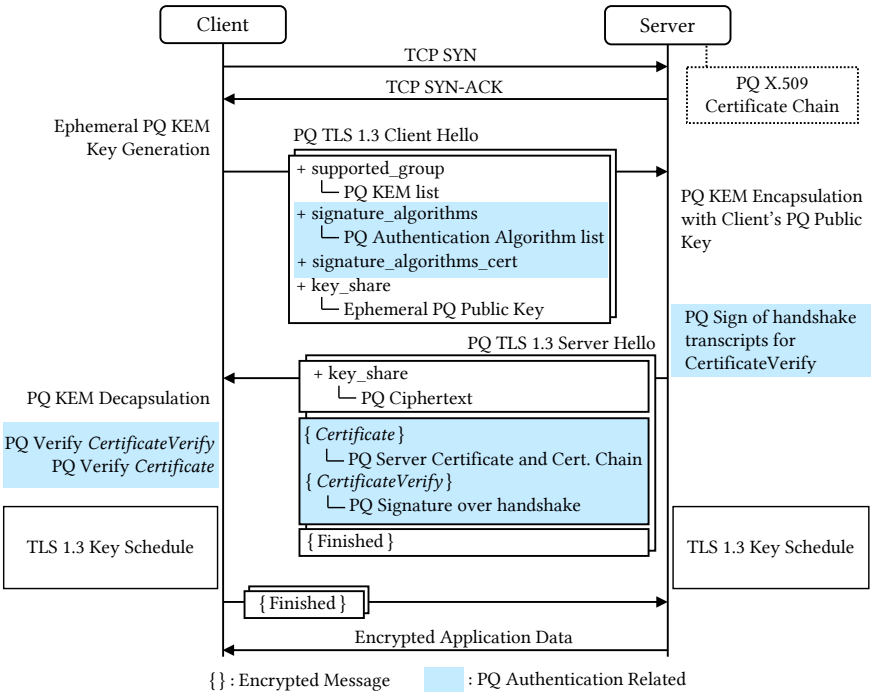

Figure 1: Post-Quantum Authentication - TLS 1.3 Handshake

alternatives for authentication, in contrast, offer public key and signature sizes from a few Kilobytes up to hundreds of Kilobytes (KB) depending on the algorithm. These increases will effectively increase the size of PQ X.509 certificates that bind public keys to online entity identities. [24, 62].

This increase in authentication data size will have a adverse effect on handshake performance within PQ TLS. As seen in Fig. 1, the client initiates the handshake by using a PQ public key as keyshare (part of the `ClientHello`), while the server uses it to perform a PQ KEM encapsulation operation before responding with the generated ciphertext in the `ServerHello` [12,44]. Part of this `ServerHello` message is also the PQ X.509 certificate chain, and the handshake transcripts for the `CertificateVerify` message which are now significantly larger in size. Recent studies [40, 54, 55] show that this triggers multiple round trips within the TLS handshake packet exchange due to small initial windows sizes pre-configured in most TCP congestion control implementations `initcwnd` (10MSS $\approx$ 14.5KB). This also increases the number of packets needed for handshakes and increased loss probability in cases of congested networks [40]. The result is a significant slowdown in secure TLS tunnel establishment.

The issue becomes even more prominent when considering the exact size of the `ServerHello` message's authentication data – especially since the number of signatures involved in TLS handshakes can vary. In modern PKI, intermediate CAs (ICA) can sign certificates for other ICAs resulting to chains of size one, two, or more trusted certificates extending to a leaf. TLS handshakes must include a signature and a public key *per certificate* in this chain, and one additional signature as part of the `CertificateVerify` message. One must also account for the cases when the Online Certificate Status Protocol (OCSP) [50] is used in TLS. In this case, an additional OCSP signature is added to the handshake to verify the revocation status of the certificates. Finally, actual X.509 Web server certificates often include Signed Certificate Timestamps (SCTs) [28, 31],

Signature Algorithm	NIST PQ Security	Single ICA (KB)	Two ICAs (KB)	Three ICAs (KB)
ECDSA *256*	-	0.77	1.10	1.44
RSA *2048*	-	2.13	2.78	3.44
Falcon *512*	Level 1	5.04	6.47	7.90
Falcon *1024*	Level 5	9.28	11.81	14.35
Dilithium *II*	Level 2	13.59	16.57	19.55
Dilithium *III*	Level 3	18.53	22.59	26.66
Dilithium *V*	Level 5	25.45	30.91	36.35
SPHINCS$^+$ 128s	Level 1	36.76	42.73	48.69
SPHINCS$^+$ 128f	Level 1	79.22	91.84	104.45

Table 1: Conventional & PQ TLS Authentication Data Size

each of which utilize one signature. The exact number of SCTs can vary; Chrome requests two to five SCTs depending on the lifetime of the certificate, while Apple requires three SCTs in case the lifetime of the certificate exceeds 180 days mark [4, 47].

In making TLS quantum safe, all the aforementioned signatures and public keys needed for authentication will make use of NIST PQ alternatives. A careful evaluation is needed to study directly the impact of these NIST Round 3 signature candidates [35]. We evaluate the variants of all security levels including Falcon and Dilithium, the two front-runners in terms of signature/key sizes and practical use in TLS. In addition, we present the authentication data produced by the most space-efficient variants from the rest candidates. We assume that each X509 certificate is in binary DER encoding [21, 22][‡], and contains 400 bytes of attribute data. Table 1 shows the accumulated authentication data per handshake in a realistic case where one extra OCSP staple and two SCTs are used (i.e., three extra signatures overall). Although NIST has announced the imminent standardisation of multiple PQ signature schemes [1], we use the same algorithm for all certificates within each chain. The advantages of using a mixed chain strategy are explored thoroughly in [41] and [55].

The PQ auth data indicate that only Falcon-512 results to an acceptable range for the most common Web TLS cases of up to 3 ICAs, while Dilithium-2 is marginally an acceptable choice for the single ICA case. In all other cases, the candidates produce, even at their lower security levels, auth data that exceed the usual `initcwnd` size of 14.5KB, that triggers additional round-trips (more than one in most cases). Thus, the alleviation of a part of the exchanged information can make more candidates feasible for use in PQ TLS without any slowdowns.

4 ICA Certificate Suppression

4.1 Approximate Membership Queries

Approximate Membership Queries (AMQs) utilize probabilistic data structures (a.k.a., "filters") to indicate whether a queried data element is part of a set. They offer extremely fast query speeds

[‡]DER encoding is also used during transfer and is $\approx 72\%$ smaller than the CRT format

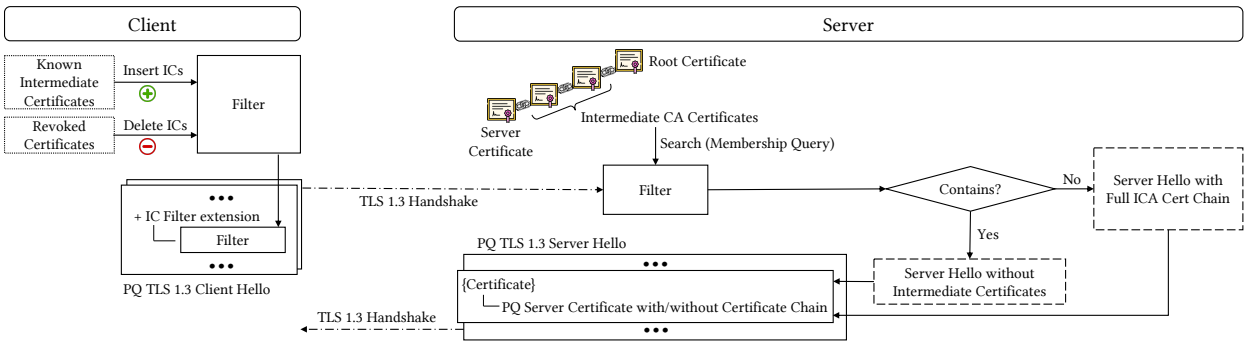

Figure 2: Client and Server-side pipeline of ICA Certificate Suppression Utilizing Approximate Membership Queries

while using far less memory than the set of data itself. In exchange, these probabilistic filters allow for a small false-positive rate ϵ, namely a small probability that a membership query may falsely indicate that the item c in question is present in the set S, when in fact it is not. The actual size of a filter (i.e., in bytes) that represents a set of n items depends on both n, and the false-positive rate ϵ. In general, the more items are added to a filter the higher the false-positive probability becomes given a specific filter size. Such probabilistic filters are widely used in networks [8] and distributed systems [56] for Ethernet switching [64], reducing space in routing tables [63], IP address matching [13], content delivery networks [30], and specifically as certificate-representing sets for certificate validation in IoT [58], and certificate revocation testing [27, 52]. While the most popular structure of this kind is the Bloom filter [7], in its basic form, it does not allow for element removal without having to rebuild the whole filter. Recently alternatives such as Cuckoo [15] or Quotient [38] filters allow for dynamic updates, namely item insertions, deletions, and changes, making AMQs more practical for real world applications.

4.2 Filter-based ICA Certificate Suppression

Fig. 2 presents an overview of the proposed IC suppression mechanism from both client and server points of view. The client maintains a list of known intermediate certificates —set S— (e.g., in a separate cache [23]), and creates an AMQ filter using them as data elements c, where $c \in S$. For the needs of our application we assume that the filter supports dynamic updates (e.g., insertions/deletions) since creating a new filter for every TLS connection or for every single-cert change would be computationally inefficient. Also, the dynamic updates simplify the deletion of revoked or expired certificates or the addition of newly discovered ICs. Subsequently, in each TLS handshake, the filter is added to the `ClientHello` message as a TLS 1.3 extension [44]. That way the client advertises to the server the ICA certs under possession without having to maintain any cross matching information between peers and their respective ICA cert chains. In addition, apart from the universal use of the same AMQ filter, no other parameters need to be communicated between the two handshake parties.

The server receives the IC filter extension and performs a high-speed membership query (lookup) for each IC in his verification path. If the ICA cert is found to be part of the set that the filter represents, then it is omitted from the `ServerHello` message response. Otherwise, the server will proceed as usual and transmit the full ICA cert chain. In both cases, the certificate chain validation process does not change. Even in the ICA cert suppression case, the client utilizes the known ICs

to complete the verification path and validate the server's leaf certificate.

In our approach, the client can advertise ICA certs to new or not-encountered-before servers by adding an additional payload to the `ClientHello` specifying the specific filter used (e.g., Quotient [38], Cuckoo [15]). The structure creation and querying are extremely fast processes with minimal computational overhead. On the other hand, due to the probabilistic nature of the utilized filter, a non-zero false-positive probability exists in each server lookup. In case of a false positive, the server would wrongfully omit the IC from the `ServerHello`, leading to a repetition of the handshake. On this repeated handshake, the client does not include the IC Suppression extension and the handshake is completed as usual. Since the false-positive probability is directly related (reversely-proportional) to the filter size ($\equiv$ handshake data overhead), filter capacity (i.e., number of ICAs to advertise) should be carefully configured to balance the tradeoff. The exact impact of this additional handshake is hard to determine without actual deployment of post-quantum authentication at scale. However, the handshake time of the false positive case will be approximate to the duration of a conventional TLS handshake d_c plus the full duration of a PQ TLS handshake d_{PQ} (which can vary depending on the chosen PQ authentication algorithm). Therefore, while the best case scenario is to achieve d_c for all PQ handshakes instead of the initial d_{PQ}, the actual large scale handshake duration of the proposed solution is $(1 - \epsilon) * d_c + \epsilon * d_{PQ}$, where ϵ is the AMQ filter's false-positive rate.

5 Evaluation

5.1 Intermediate Certificates in the Wild

In designing the IC filter, we start by examining the exact number of active ICA certificates in the wild. First, we examined Firefox's ICA Preload list [16] which includes all non-revoked, non-expired ICA certs used in Web chains as exported from the Common CA Database (CCADB) [33]. On June 21, 2022, this list contained 1400 distinct Web ICA certificates. However, prior investigation of the chains in 1M top domain lists lead to significantly smaller numbers (e.g., [23] found ≈375 distinct ICA certs in Cisco's Umbrella).

Thus, in an effort to verify this number, we independently investigated the ICs and cert chains returned from the most popular sites utilizing the Tranco list [29] which implements a research-oriented manipulation-hardened ranking of web domains. We manually collected ICs from the top

Tranco List	Unique	Total # of	Certificate Chain Sizes Observed (%)				
Dates	ICAs	Servers	0 ICAs	1 ICA	2 ICAs	3 ICAs	>3 ICAs
Jan. '22	220	10K	30.8	35.6	24.1	9.4	0.1
Feb. '22	236	10K	14.4	43.5	30.2	11.8	0.1
Mar. '22	228	10K	13.3	44.8	30.2	11.6	0.1
Apr. '22	231	10K	13.7	44.7	30.0	11.5	0.1
May '22	224	10K	19.7	41.6	27.5	11.0	0.2
Jun. '22	245	10K	24.1	39.1	26.5	10.1	0.2

Table 2: Certificate chain data as observed in the Tranco Top Sites list (Top 10K entries, Jan.'22-May'22).

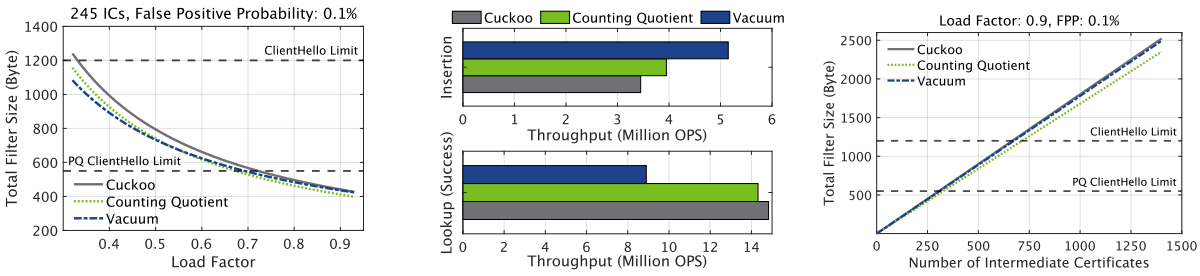

Figure 3: Feasibility of AMQ Filters: (left) size vs load factor, (center) throughput, (right) size vs certificate capacity

10K domains as ranked over a period of 6 months, and for servers listening to the 443 port (TLS). These results are shown in Table 2. We observe that over 80% of the examined servers include at least one ICA, which underlines the need for ICA cert suppression in the PQ TLS era. The number of distinct ICs —that showcases the IC filter capacity needs— varies between 220 and 245 certificates.

5.2 Filter Design and Overhead

Following our observations above, the IC filter capacity should lie between 220 and 1400 intermediate certificates. Also, the filter design is limited by the available space at the `ClientHello` message that should not exceed the $\approx 1.4KB$ mark. In the current PKI, `ClientHello` messages that carry X25519 keyshares amount to approximately 250 bytes since the public key is only 32 bytes. This leaves $\approx 1.2KB$ of available space for our filter's size. However, since in a PQ TLS scenario we would utilize a PQ KEM algorithm [35], the `ClientHello` message will contain a larger PQ public key —e.g., 699 bytes for NTRU-HPS-509 [11], or 672 bytes for Lightsaber [6]— making the message $\approx$ 890 to 917 bytes. This further limits our filter's maximum size to $\approx$ 550 bytes.

Naturally, the above calculations concern the Linux default TCP initial congestion window `initcwnd` of 10 MSS that is used by a significant majority [48, 49]. However, recent studies have showed significant customization of `initcwnd`, as well as differentiation of the TCP receive window among different operating systems [49]. This diversity can impact our proposed solution. First, an increased initial window can lead to either the inclusion of more ICAs inside the AMQ filter or to a lower false-positive rate, resulting to a more stable mechanism operation. On the other hand, when the initial windows are large, the long certificates of PQ schemes do not exceed them, and therefore no round-trips are added to the handshakes. In this case, the initiator of the handshake can omit the IC Filter extension altogether. Therefore, the initial window setting can act as an additional input parameter for the initiator of the handshake towards deciding whether to use the proposed framework, or not. Note that any imprudent universal increase of the `initcwnd` can add to the loss rate in cases of low bandwidth or congested connections resulting to the opposite of the desired effect. A detailed discussion of the `initcwnd` impact in PQ TLS can be found in [54, § 5].

Next, we investigate the feasibility of our filter-based approach by examining AMQ structure variants that allow for dynamic updates, namely Cuckoo [15], Vacuum [59], and Counting Quotient [39] filters. First, we evaluate the filter's load factor for a targeted capacity of 245 ICs and FFP of 0.1%. To achieve a space utilization that will enable PQ IC suppression, load factors should remain above 75% in all cases (Fig.3-left). Thus, we will use a load factor of 90% where all variants perform similarly. Next, we evaluate the filter's throughput in terms of IC insertions and

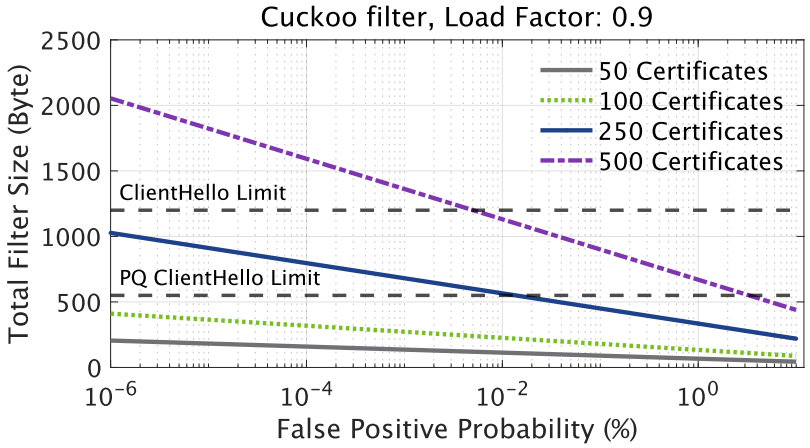

Figure 4: IC Suppression Extension Size vs FPP

most importantly IC queries (Fig. 3-center). We find that all AMQ structures can handle millions of lookups in seconds which fits our case since servers often handle this amount of handshakes simultaneously. Regarding the IC capacity, Fig. 3-right shows the structure's size as a function of represented ICs. All structures manage to stay below 550 bytes and at the same time hold over 300 ICs which is within the required number especially in the popular domain space. Finally, Fig.4 evaluates the filter's size against the targeted false positive probability (FPP). This reversely-proportional relation between FPP and `ClientHello` size can provide an adjustable parameter for different TLS use cases. For instance, an app client that communicates with a small set of peers (e.g., service mesh cases) can aim for a small FPP with less advertised ICs.

5.3 IC Suppression Impact Estimation

In this subsection, we quantify the impact of filter-based ICA Suppression in a real-world scenario, namely in the context of the TLS-reliant HTTPS protocol. We simulate the browsing behaviour of a typical user visiting domains from the Tranco list [29] (Top 1M) following the model found in [9]. We used the lower bound of all the model parameters in [9] per Table 2. In short, the simulated user visits Tranco domains following a Zipf-like distribution (exponent=1.9), views pages with a Pareto distribution (exp=0.7), and leaves pages with a Pareto distribution (exp=2.5). We utilize the Cuckoo filter with a 0.9 load factor, 0.1% FFP, and insert the set of 245 ICAs (Jun. '22 Tranco list - Tab. 2). The experimental setup consisted of a local host running Ubuntu 18.04 in x86_64 architecture, equipped with an Intel i7-8665u that utilized two cores at 1.9 GHz each, and 4 GBs of RAM. The default TCP initial congestion window of 10 MSS was initially used for all experiments.

In each visited domain, the simulator performs a TLS handshake with the web page and with any embedded HTTPS content from third parties available. For each TLS handshake we collect the certificate chain, and measure the authentication data, the round trip time to the server, and the time to the first byte (TTFB) in content retrieval. Next, for each TLS handshake we simulate the proposed framework's server action by querying the retrieved ICA certs against the IC filter and mark the outcome of the three possibilities: (a) ICA certs are found (successful ICA suppression), (b) ICA certs are not found (normal handshake), and (c) we observe a false positive (handshake is repeated with no suppression).

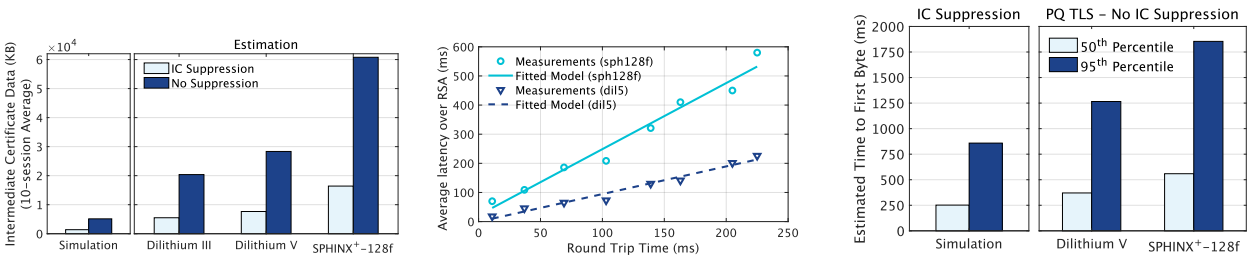

Figure 5: IC Suppression Impact Estimation: (left) IC data exchanged, (center) PQ authentication induced latency, (right) time to first byte

We conducted 10 runs of our simulator, where the user visited 200 domains every time. Fig. 5-left shows the average ICA cert data volume in a single browsing session with and without ICA suppression. The introduction of the framework led to an $\approx 73\%$ decrease of the exchanged ICA cert data. Note that, on average, in each browsing session the simulator loaded secure content from ≈ 1950 unique destinations, while we observed common ICA certs in a rate that varied between $69-74\%$ with an average of 2.3 false positives per run. Also, we present an extrapolation of the same metric for the PQ TLS cases where Dilithium III/V and SPHINCS$^+$ 128f are used. On average, we estimate that, in a similar environment, the ICA suppression mechanism can save $\approx 15MB$ in exchanged PQ authentication data for Dilithium III, while for SPHINCS$^+$ 128f, $\approx 45MB$ are conserved.

Finally, we estimate the impact that our proposed framework has on the average user by calculating the TTFB for different scenarios. When ICA suppression is considered, we add the average filter lookup time to each measured TTFB, while in the case of a false-positive the observed TTFB is doubled. We assume that the IC filter was already created therefore the construction time is not amortized over TTFB measurements. To estimate the TTFB in case of PQ authentication, we performed individual measurements to extract the additional latency over RSA 2048 following the experimental setup suggested in [55]. Since additional delays are a function of the RTTs to the remote servers [54] we present in Fig. 5-center the average latency induced by Dilithium V and SPHINCS$^+$ 128f over RSA 2048 and calculate a latency model based on the line of best-fit (linear regression). Using these models and the measured RTTs from the simulation, we estimate the TTFB for the PQ authentication cases. Fig. 5-left shows these results (TTFB measurements from all 10 sessions), and highlights potential latency gains to the user from the use of our framework. The measurement from our simulation is an upper limit on the TTFB since most of the handshakes we observed did not include any roundtrips to fetch the auth data. Note that the fast query speeds and the small number of false positives added negligible delay to the ICA suppression emulated TTFB. Since this phenomenon produces the majority of observed latency in Dilithium V and SPHINX$^+$, the use of ICA suppression can reduce the number of roundtrips within these PQ TLS handshakes, and speedup the TTFB for the user by hundreds of milliseconds. Thus, with our approach, candidates such as SPHINX can become more appealing for use in TLS, while others like Dilithium II/III will present performance on par with RSA.

6 Security Considerations

Since the `ClientHello` message is sent in cleartext on the wire as part of the TLS handshake [44], any passive observer can have access to the IC filter that is added as an extension. Thus, the proposed framework creates unencrypted signals that could be used to identify which ICA certs are known, increasing the effectiveness of client fingerprinting as noted also in [57]. While this indeed constitutes a partial loss of privacy, no information regarding the actual identity of the two parties is leaked. A solution to this drawback is the use of public key encryption to encrypt the `ClientHello` message as suggested in the IETF draft-ietf-tls-esni [46]. In addition, using the proposed mechanism and therefore sending the ICA filter to only known, and thus trusted peers, may further alleviate privacy concerns. Regarding unknown servers, the use of carefully curated and universal ICA filters can accordingly mitigate loss of privacy. This is because peer history of encountered ICAs will not be shared in this case, resulting to a consequent reduction of our mechanism's performance due to the loss of targeted ICA advertisement. The use of the same ICA suppression mechanism in client authentication (i.e., mTLS), does not present the same leakage since in TLS 1.3 all handshake messages after the `ServerHello` are encrypted anyway.

In addition, no security implications arise from the validation of the cert chain as ICAs are still not trusted by default. Revocation checking practices at the client are not altered by the proposed ICA suppression framework, that also allows for dynamic update of the IC filter contents. Finally, Mozilla's recent Intermediate CA Preloading framework [61] also attests to the safety of utilizing already known ICA certs.

7 Conclusion

In this work, we underline the challenge of introducing PQ authentication to TLS handshakes (larger PQ certs means unacceptable latency) and propose an ICA suppression mechanism that reduces the exchanged information, preventing excessive round-trip induced delays. We rely on approximate membership query filters to indicate our set of known ICA certificates to the TLS peer, towards the omission of the latter from the handshake. Our results indicate a reduction of $\approx 73\%$ in exchanged data volumes and speedups in the hundreds of milliseconds in the case of PQ TLS.

In future work, we plan to evaluate the ICA suppression performance in non-Web-based environments (e.g., IoT, mobile devices), and utilize targeted advertisement of specific ICAs to specific peers through adaptive filter construction.

Acknowledgments

We would like to thank Rob Johnson from VMware Research for his valuable feedback on AMQ structures. We would also like to thank Panos Kampanakis from Amazon Web Services for various discussions on intermediate certificate suppression. The work of Michael Devetsikiotis was supported by a Cisco research grant.

References

[1] G. Alagic, D. Apon, D. Cooper, Q. Dang, T. Dang, J. Kelsey, J. Lichtinger, Y.-K. Liu, C. Miller, D. Moody, et al. Status report on the third round of the nist post-quantum cryptography standardization process. *US Department of Commerce, NIST*, 2022. `https://nvlpubs.nist.gov/nistpubs/ir/2022/NIST.IR.8413.pdf`.

[2] ANSI. The Elliptic Curve Key Agreement and Key Transport Protocols, September 1999. American National Standards Institute, X9-Financial Services.

[3] ANSI. ANSI X9.62, Public Key Cryptography For The Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA), September 2005. American National Standards Institute, X9-Financial Services.

[4] Apple. Apple's Certificate Transparency policy, Mar. 2021. `https://support.apple.com/en-us/HT205280`.

[5] J.-P. Aumasson, D. J. Bernstein, C. Dobraunig, M. Eichlseder, S. Fluhrer, S.-L. Gazdag, A. Hülsing, P. Kampanakis, S. Kölbl, T. Lange, et al. SPHINCS+ - Submission to the 2nd round of the NIST post-quantum project. `https://sphincs.org/data/sphincs+-round2-specification.pdf`, 2019. Specification document (part of the submission package).

[6] A. Basso, J. M. B. Mera, J.-P. D'Anvers, A. Karmakar, S. S. Roy, M. Van Beirendonck, and F. Vercauteren. Saber: Mod-lwr based kem (round 3 submission), 2020. `https://www.esat.kuleuven.be/cosic/pqcrypto/saber/files/saberspecround3.pdf`.

[7] B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, July 1970.

[8] A. Broder and M. Mitzenmacher. Network applications of bloom filters: A survey. *Internet mathematics*, 1(4):485–509, 2004.

[9] S. Burklen, P. J. Marron, S. Fritsch, and K. Rothermel. User centric walk: An integrated approach for modeling the browsing behavior of users on the web. In *38th Annual Simulation Symposium*, pages 149–159. IEEE, 2005.

[10] C.-l. Chan, R. Fontugne, K. Cho, and S. Goto. Monitoring TLS adoption using backbone and edge traffic. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 208–213. IEEE, 2018.

[11] C. Chen, O. Danba, J. Hoffstein, A. Hülsing, J. Rijneveld, J. M. Schanck, P. Schwabe, W. Whyte, and Z. Zhang. Ntru algorithm specifications and supporting documentation. 2020. `https://ntru.org/`.

[12] E. Crockett, C. Paquin, and D. Stebila. Prototyping post-quantum and hybrid key exchange and authentication in tls and ssh. In *NIST 2nd Post-Quantum Cryptography Standardization Conference 2019*, 2019.

[13] S. Dharmapurikar, P. Krishnamurthy, and D. E. Taylor. Longest prefix matching using bloom filters. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, page 201–212, New York, NY, USA, 2003. Association for Computing Machinery.

[14] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé. CRYSTALS-Dilithium Algorithm Specifications and Supporting Documentation. `https://pq-crystals.org/dilithium/resources.shtml`, 2018. Submission to round 2 of the NIST post-quantum project.

[15] B. Fan, D. G. Andersen, M. Kaminsky, and M. D. Mitzenmacher. Cuckoo filter: Practically better than bloom. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, pages 75–88, 2014.

[16] Firefox. Firefox's ica preload list. `https://wiki.mozilla.org/CA/Intermediate_Certificates`.

[17] P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang. Falcon: Fast-Fourier lattice-based compact signatures over NTRU. `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`, 2018. Specification v1.1.

[18] Google. Google Transparency Report - HTTPS encryption on the web. `https://transparencyreport.google.com/https/overview`, 2020. Web page. Accessed 2020-06-19.

[19] P. E. Hoffman. SMTP Service Extension for Secure SMTP over Transport Layer Security. RFC 3207, Feb. 2002.

[20] R. Holz, J. Hiller, J. Amann, A. Razaghpanah, T. Jost, N. Vallina-Rodriguez, and O. Hohlfeld. Tracking the deployment of tls 1.3 on the web: A story of experimentation and centralization. *SIGCOMM Comput. Commun. Rev.*, 50(3):3–15, July 2020.

[21] K. Igoe and D. Stebila. X.509v3 Certificates for Secure Shell Authentication. RFC 6187, Mar. 2011.

[22] International Telecommunications Union (ITU-T). ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER). `https://www.itu.int/rec/T-REC-X.690-202102-I/en`.

[23] P. Kampanakis and M. Kallitsis. Faster post-quantum tls handshakes without intermediate ca certificates. In *The 6th International Symposium on Cyber Security, Cryptology and Machine Learning (CSCML 2022), ICMC 2021*, 2022.

[24] P. Kampanakis and D. Sikeridis. Two pq signature use-cases: Non-issues, challenges and potential solutions. Cryptology ePrint Archive, Paper 2019/1276, 2019. `https://eprint.iacr.org/2019/1276`.

[25] P. Kampanakis, D. Stebila, M. Friedl, T. Hansen, and D. Sikeridis. Post-quantum public key algorithms for the Secure Shell (SSH) protocol. Internet-Draft draft-kampanakis-curdle-pq-ssh-00, Internet Engineering Task Force, Oct. 2020. Work in Progress.

[26] P. Kotzias, A. Razaghpanah, J. Amann, K. G. Paterson, N. Vallina-Rodriguez, and J. Caballero. Coming of age: A longitudinal study of tls deployment. In *Proceedings of the Internet Measurement Conference 2018*, pages 415–428. ACM, 2018.

[27] J. Larisch, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson. Crlite: A scalable system for pushing all tls revocations to all browsers. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 539–556. IEEE, 2017.

[28] B. Laurie, A. Langley, and E. Kasper. Certificate Transparency. RFC 6962, June 2013.

[29] V. Le Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen. Tranco: A research-oriented top sites ranking hardened against manipulation. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium*, NDSS 2019, Feb. 2019.

[30] B. M. Maggs and R. K. Sitaraman. Algorithmic nuggets in content delivery. *ACM SIGCOMM Computer Communication Review*, 45(3):52–66, 2015.

[31] S. Meiklejohn, J. DeBlasio, D. O'Brien, C. Thompson, K. Yeo, and E. Stark. Sok: Sct auditing in certificate transparency, 2022.

[32] K. Moriarty, B. Kaliski, J. Jonsson, and A. Rusch. PKCS# 1: RSA cryptography specifications version 2.2. *Internet Engineering Task Force, Request for Comments*, 8017, 2016.

[33] Mozilla. Common CA Database (CCADB), Feb. 2022. `https://ccadb-public.secure.force.com/mozilla/MozillaIntermediateCertsCSVReport`.

[34] D. Naylor, A. Finamore, I. Leontiadis, Y. Grunenberger, M. Mellia, M. Munafò, K. Papagiannaki, and P. Steenkiste. The cost of the S in HTTPS. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, pages 133–140. ACM, 2014.

[35] NIST. Post-Quantum Cryptography Round 3 Submissions, 2020. `https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions`.

[36] D. Ott, D. Moreau, and M. Gaur. Planning for cryptographic readiness in an era of quantum computing advancement. In *ICISSP*, pages 491–498, 2022.

[37] D. Ott, C. Peikert, et al. Identifying research challenges in post quantum cryptography migration and cryptographic agility. *arXiv preprint arXiv:1909.07353*, 2019.

[38] P. Pandey, M. A. Bender, R. Johnson, and R. Patro. A general-purpose counting filter: Making every bit count. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, page 775–787, New York, NY, USA, 2017. Association for Computing Machinery.

[39] P. Pandey, M. A. Bender, R. Johnson, and R. Patro. A general-purpose counting filter: Making every bit count. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, page 775–787, New York, NY, USA, 2017. Association for Computing Machinery.

[40] C. Paquin, D. Stebila, and G. Tamvada. Benchmarking post-quantum cryptography in tls. In *International Conference on Post-Quantum Cryptography*, pages 72–91. Springer, 2020.

[41] S. Paul, Y. Kuzovkova, N. Lahr, and R. Niederhagen. Mixed certificate chains for the transition to post-quantum authentication in tls 1.3. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '22, page 727–740, New York, NY, USA, 2022. Association for Computing Machinery.

[42] J. Proos and C. Zalka. Shor's discrete logarithm quantum algorithm for elliptic curves. *Quantum Info. Comput.*, 3(4):317–344, July 2003.

[43] A. Razaghpanah, A. A. Niaki, N. Vallina-Rodriguez, S. Sundaresan, J. Amann, and P. Gill. Studying tls usage in android apps. In *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies*, pages 350–362, 2017.

[44] E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, Aug. 2018.

[45] E. Rescorla, R. Barnes, and H. Tschofenig. Compact TLS 1.3. Internet-Draft draft-rescorla-tls-ctls-04, Internet Engineering Task Force, Mar. 2020. Work in Progress.

[46] E. Rescorla, K. Oku, N. Sullivan, and C. A. Wood. TLS Encrypted Client Hello. Internet-Draft draft-ietf-tls-esni-14, Internet Engineering Task Force, Feb. 2022. Work in Progress.

[47] I. Ristić. Certificate transparency policies diverge after apple's update, 2021. `https://www.hardenize.com/blog/certificate-transparency-policies-diverge-with-apples-update?utm_source=fd&utm_medium=email&utm_campaign=nl76`.

[48] J. Rüth, C. Bormann, and O. Hohlfeld. Large-scale scanning of tcp's initial window. In *Proceedings of the 2017 Internet Measurement Conference*, pages 304–310, 2017.

[49] J. Rüth and O. Hohlfeld. Demystifying tcp initial window configurations of content distribution networks. In *2018 Network Traffic Measurement and Analysis Conference (TMA)*, pages 1–8. IEEE, 2018.

[50] S. Santesson, M. Myers, R. Ankney, A. Malpani, S. Galperin, and D. C. Adams. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. RFC 6960, June 2013.

[51] P. Schwabe, D. Stebila, and T. Wiggers. Post-quantum tls without handshake signatures. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, page 1461–1480, New York, NY, USA, 2020. Association for Computing Machinery.

[52] X. Shi, S. Shi, M. Wang, J. Kaunisto, and C. Qian. On-device iot certificate revocation checking with small memory and low latency. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 1118–1134, New York, NY, USA, 2021. Association for Computing Machinery.

[53] P. W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM J. on Computing*, 26(5):1484–1509, 1997.

[54] D. Sikeridis, P. Kampanakis, and M. Devetsikiotis. Assessing the overhead of post-quantum cryptography in tls 1.3 and ssh. In *Proceedings of the 16th International Conference on emerging Networking EXperiments and Technologies*, pages 149–156, 2020.

[55] D. Sikeridis, P. Kampanakis, and M. Devetsikiotis. Post-Quantum Authentication in TLS 1.3: A Performance Study. In *Network and Distributed Systems Security (NDSS) Symposium 2020 23-26 February 2020, San Diego, CA, USA*. The Internet Society, 2020.

[56] S. Tarkoma, C. E. Rothenberg, and E. Lagerspetz. Theory and practice of bloom filters for distributed systems. *IEEE Communications Surveys & Tutorials*, 14(1):131–155, 2011.

[57] M. Thomson. Suppressing Intermediate Certificates in TLS. Internet-Draft draft-thomson-tls-sic-00, Internet Engineering Task Force, Mar. 2019. Work in Progress.

[58] M. Wang, C. Qian, X. Li, S. Shi, and S. Chen. Collaborative validation of public-key certificates for iot by distributed caching. *IEEE/ACM Transactions on Networking*, 29(1):92–105, 2020.

[59] M. Wang, M. Zhou, S. Shi, and C. Qian. Vacuum filters: More space-efficient and faster replacement for bloom and cuckoo filters. *Proc. VLDB Endow.*, 13(2):197–210, Oct. 2019.

[60] B. Westerbaan. Sizing Up Post-Quantum Signatures, Nov. 2021. `https://blog.cloudflare.com/sizing-up-post-quantum-signatures/`.

[61] M. Wiki. Intermediate CA Preloading. `https://wiki.mozilla.org/Security/CryptoEngineering/Intermediate_Preloading`, 2020.

[62] P. E. Yee. Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 6818, Jan. 2013.

[63] M. Yu, A. Fabrikant, and J. Rexford. Buffalo: Bloom filter forwarding architecture for large organizations. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 313–324, 2009.

[64] D. Zhou, B. Fan, H. Lim, M. Kaminsky, and D. G. Andersen. Scalable, high performance ethernet forwarding with cuckooswitch. In *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*, pages 97–108, 2013.