Stream: Internet Engineering Task Force (IETF)
RFC: 8901
Category: Informational
Published: September 2020
ISSN: 2070-1721
Authors: S. Huque, *Salesforce*; P. Aras, *Salesforce*; J. Dickinson, *Sinodun IT*; J. Vcelak, *NS1*; D. Blacka, *Verisign*

RFC 8901
Multi-Signer DNSSEC Models

Abstract

Many enterprises today employ the service of multiple DNS providers to distribute their authoritative DNS service. Deploying DNSSEC in such an environment may present some challenges, depending on the configuration and feature set in use. In particular, when each DNS provider independently signs zone data with their own keys, additional key-management mechanisms are necessary. This document presents deployment models that accommodate this scenario and describes these key-management requirements. These models do not require any changes to the behavior of validating resolvers, nor do they impose the new key-management requirements on authoritative servers not involved in multi-signer configurations.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are candidates for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc8901.

Copyright Notice

Table of Contents

1. Introduction and Motivation

Many enterprises today employ the service of multiple Domain Name System (DNS) [RFC1034] [RFC1035] providers to distribute their authoritative DNS service. This is primarily done for redundancy and availability, and it allows the DNS service to survive a complete, catastrophic failure of any single provider. Additionally, enterprises or providers occasionally have requirements that preclude standard zone-transfer techniques [RFC1995][RFC5936]: either nonstandardized DNS features are in use that are incompatible with zone transfer, or operationally a provider must be able to (re-)sign DNS records using their own keys. This document outlines some possible models of DNSSEC [RFC4033] [RFC4034] [RFC4035] deployment in such an environment.

This document assumes a reasonable level of familiarity with DNS operations and protocol terms. Much of the terminology is explained in further detail in "DNS Terminology" [RFC8499].

2. Deployment Models

If a zone owner can use standard zone-transfer techniques, then the presence of multiple providers does not require modifications to the normal deployment models. In these deployments, there is a single signing entity (which may be the zone owner, one of the providers, or a separate entity), while the providers act as secondary authoritative servers for the zone.

Occasionally, however, standard zone-transfer techniques cannot be used. This could be due to the use of nonstandard DNS features or the operational requirements of a given provider (e.g., a provider that only supports "online signing"). In these scenarios, the multiple providers each act like primary servers, independently signing data received from the zone owner and serving it to DNS queriers. This configuration presents some novel challenges and requirements.

2.1. Multiple-Signer Models

In this category of models, multiple providers each independently sign and serve the same zone. The zone owner typically uses provider-specific APIs to update zone content identically at each of the providers and relies on the provider to perform signing of the data. A key requirement here is to manage the contents of the DNSKEY and Delegation Signer (DS) RRsets in such a way that validating resolvers always have a viable path to authenticate the DNSSEC signature chain, no matter which provider is queried. This requirement is achieved by having each provider import the public Zone Signing Keys (ZSKs) of all other providers into their DNSKEY RRsets.

These models can support DNSSEC even for the nonstandard features mentioned previously, if the DNS providers have the capability of signing the response data generated by those features. Since these responses are often generated dynamically at query time, one method is for the provider to perform online signing (also known as on-the-fly signing). However, another possible approach is to precompute all the possible response sets and associated signatures and then algorithmically determine at query time which response set and signature need to be returned.

In the models presented, the function of coordinating the DNSKEY or DS RRset does not involve the providers communicating directly with each other. Feedback from several commercial managed-DNS providers indicates that they may be unlikely to directly communicate, since they typically have a contractual relationship only with the zone owner. However, if the parties involved are agreeable, it may be possible to devise a protocol mechanism by which the providers directly communicate to share keys. Details of such a protocol are deferred to a future specification document, should there be interest.

In the descriptions below, the Key Signing Key (KSK) and Zone Signing Key (ZSK) correspond to the definitions in [RFC8499], with the caveat that the KSK not only signs the zone apex DNSKEY RRset but also serves as the Secure Entry Point (SEP) into the zone.

2.1.1. Model 1: Common KSK Set, Unique ZSK Set per Provider

- The zone owner holds the KSK set, manages the DS record set, and is responsible for signing the DNSKEY RRset and distributing it to the providers.
- Each provider has their own ZSK set, which is used to sign data in the zone.
- The providers have an API that the zone owner uses to query the ZSK public keys and insert a combined DNSKEY RRset that includes the ZSK sets of each provider and the KSK set, signed by the KSK.
- Note that even if the contents of the DNSKEY RRset do not change, the zone owner needs to periodically re-sign it as signature expiration approaches. The provider API is also used to thus periodically redistribute the refreshed DNSKEY RRset.
- Key rollovers need coordinated participation of the zone owner to update the DNSKEY RRset (for KSK or ZSK) and the DS RRset (for KSK).
- (One specific variant of this model that may be interesting is a configuration in which there is only a single provider. A possible use case for this is where the zone owner wants to outsource the signing and operation of their DNS zone to a single third-party provider but still control the KSK, so that they can authorize and/or revoke the use of specific zone signing keys.)

2.1.2. Model 2: Unique KSK Set and ZSK Set per Provider

- Each provider has their own KSK and ZSK sets.
- Each provider offers an API that the zone owner uses to import the ZSK sets of the other providers into their DNSKEY RRset.
- The DNSKEY RRset is signed independently by each provider using their own KSK.
- The zone owner manages the DS RRset located in the parent zone. This is comprised of DS records corresponding to the KSKs of each provider.
- Key rollovers need coordinated participation of the zone owner to update the DS RRset (for KSK) and the DNSKEY RRset (for ZSK).

3. Validating Resolver Behavior

The central requirement for both of the multiple-signer models (Section 2.1) is to ensure that the ZSKs from all providers are present in each provider's apex DNSKEY RRset and vouched for by either the single KSK (in Model 1) or each provider's KSK (in Model 2.) If this is not done, the following situation can arise (assuming two providers, A and B):

- The validating resolver follows a referral (i.e., secure delegation) to the zone in question.
- It retrieves the zone's DNSKEY RRset from one of Provider A's nameservers, authenticates it against the parent DS RRset, and caches it.
- At some point in time, the resolver attempts to resolve a name in the zone while the DNSKEY RRset received from Provider A is still viable in its cache.
- It queries one of Provider B's nameservers to resolve the name and obtains a response that is signed by Provider B's ZSK, which it cannot authenticate because this ZSK is not present in its cached DNSKEY RRset for the zone that it received from Provider A.
- The resolver will not accept this response. It may still be able to ultimately authenticate the name by querying other nameservers for the zone until it elicits a response from one of Provider A's nameservers. But it has incurred the penalty of additional round trips with other nameservers, with the corresponding latency and processing costs. The exact number of additional round trips depends on details of the resolver's nameserver-selection algorithm and the number of nameservers configured at Provider B.
- It may also be the case that a resolver is unable to provide an authenticated response, because it gave up after a certain number of retries or a certain amount of delay; or it is possible that downstream clients of the resolver that originated the query timed out waiting for a response.

Hence, it is important that the DNSKEY RRset at each provider is maintained with the active ZSKs of all participating providers. This ensures that resolvers can validate a response no matter which provider's nameservers it came from.

Details of how the DNSKEY RRset itself is validated differ. In Model 1 (Section 2.1.1), one unique KSK managed by the zone owner signs an identical DNSKEY RRset deployed at each provider, and the signed DS record in the parent zone refers to this KSK. In Model 2 (Section 2.1.2), each provider has a distinct KSK and signs the DNSKEY RRset with it. The zone owner deploys a DS RRset at the parent zone that contains multiple DS records, each referring to a distinct provider's KSK. Hence, it does not matter which provider's nameservers the resolver obtains the DNSKEY RRset from; the signed DS record in each model can authenticate the associated KSK.

4. Signing-Algorithm Considerations

DNS providers participating in multi-signer models need to use a common DNSSEC signing algorithm (or a common set of algorithms if several are in use). This is because the current specifications require that if there are multiple algorithms in the DNSKEY RRset, then RRsets in

the zone need to be signed with at least one DNSKEY of each algorithm, as described in [RFC4035], Section 2.2. If providers employ distinct signing algorithms, then this requirement cannot be satisfied.

5. Authenticated-Denial Considerations

Authenticated denial of existence enables a resolver to validate that a record does not exist. For this purpose, an authoritative server presents, in a response to the resolver, signed NSEC (Section 3.1.3 of [RFC4035]) or NSEC3 (Section 7.2 of [RFC5155]) records that provide cryptographic proof of this nonexistence. The NSEC3 method enhances NSEC by providing opt-out for signing insecure delegations and also adds limited protection against zone-enumeration attacks.

An authoritative server response carrying records for authenticated denial is always self-contained, and the receiving resolver doesn't need to send additional queries to complete the proof of denial. For this reason, no rollover is needed when switching between NSEC and NSEC3 for a signed zone.

Since authenticated-denial responses are self-contained, NSEC and NSEC3 can be used by different providers to serve the same zone. Doing so, however, defeats the protection against zone enumeration provided by NSEC3 (because an adversary can trivially enumerate the zone by just querying the providers that employ NSEC). A better configuration involves multiple providers using different authenticated denial-of-existence mechanisms that all provide zone-enumeration defense, such as precomputed NSEC3, NSEC3 white lies [RFC7129], NSEC black lies [BLACKLIES], etc. Note, however, that having multiple providers offering different authenticated-denial mechanisms may impact how effectively resolvers are able to make use of the caching of negative responses.

5.1. Single Method

Usually, the NSEC and NSEC3 methods are used exclusively (i.e., the methods are not used at the same time by different servers). This configuration is preferred, because the behavior is well defined and closest to current operational practice.

5.2. Mixing Methods

Compliant resolvers should be able to validate zone data when different authoritative servers for the same zone respond with different authenticated-denial methods, because this is normally observed when NSEC and NSEC3 are being switched or when NSEC3PARAM is updated.

Resolver software may, however, be designed to handle a single transition between two authenticated denial configurations more optimally than a permanent setup with mixed authenticated-denial methods. This could make caching on the resolver side less efficient, and the authoritative servers may observe a higher number of queries. This aspect should be considered especially in the context of "Aggressive Use of DNSSEC-Validated Cache" [RFC8198].

In case all providers cannot be configured with the same authenticated-denial mechanism, it is recommended to limit the distinct configurations to the lowest number feasible.

Note that NSEC3 configuration on all providers with different NSEC3PARAM values is considered a mixed setup.

6. Key Rollover Considerations

The multiple-signer (Section 2.1) models introduce some new requirements for DNSSEC key rollovers. Since this process necessarily involves coordinated actions on the part of providers and the zone owner, one reasonable strategy is for the zone owner to initiate key-rollover operations. But other operationally plausible models may also suit, such as a DNS provider initiating a key rollover and signaling their intent to the zone owner in some manner. The mechanism to communicate this intent could be some secure out-of-band channel that has been agreed upon, or the provider could offer an API function that could be periodically polled by the zone owner.

For simplicity, the descriptions in this section assume two DNS providers. They also assume that KSK rollovers employ the commonly used Double-Signature KSK rollover method and that ZSK rollovers employ the Pre-Publish ZSK rollover method, as described in detail in [RFC6781]. With minor modifications, they can be easily adapted to other models, such as Double-DS KSK rollover or Double-Signature ZSK rollover, if desired. Key-use timing should follow the recommendations outlined in [RFC6781], but taking into account the additional operations needed by the multi-signer models. For example, "time to propagate data to all the authoritative servers" now includes the time to import the new ZSKs into each provider.

6.1. Model 1: Common KSK, Unique ZSK per Provider

- Key Signing Key Rollover: In this model, the two managed-DNS providers share a common KSK (public key) in their respective zones, and the zone owner has sole access to the private key portion of the KSK. To initiate the rollover, the zone owner generates a new KSK and obtains the DNSKEY RRset of each DNS provider using their respective APIs. The new KSK is added to each provider's DNSKEY RRset, and the RRset is re-signed with both the new and the old KSK. This new DNSKEY RRset is then transferred to each provider. The zone owner then updates the DS RRset in the parent zone to point to the new KSK and, after the necessary DS record TTL period has expired, proceeds with updating the DNSKEY RRset to remove the old KSK.
- Zone Signing Key Rollover: In this model, each DNS provider has separate Zone Signing Keys. Each provider can choose to roll their ZSK independently by coordinating with the zone owner. Provider A would generate a new ZSK and communicate their intent to perform a rollover (note that Provider A cannot immediately insert this new ZSK into their DNSKEY RRset, because the RRset has to be signed by the zone owner). The zone owner obtains the new ZSK from Provider A. It then obtains the current DNSKEY RRset from each provider (including Provider A), inserts the new ZSK into each DNSKEY RRset, re-signs the DNSKEY RRset, and sends it back to each provider for deployment via their respective key-management APIs. Once the necessary time period has elapsed (i.e., all zone data has been re-signed by the new ZSK and propagated to all authoritative servers for the zone, plus the maximum zone-TTL value of any of the data in the zone that has been signed by the old ZSK),

Provider A and the zone owner can initiate the next phase of removing the old ZSK and re-signing the resulting new DNSKEY RRset.

6.2. Model 2: Unique KSK and ZSK per Provider

- Key Signing Key Rollover: In Model 2, each managed-DNS provider has their own KSK. A KSK roll for Provider A does not require any change in the DNSKEY RRset of Provider B but does require co-ordination with the zone owner in order to get the DS record set in the parent zone updated. The KSK roll starts with Provider A generating a new KSK and including it in their DNSKEY RRSet. The DNSKey RRset would then be signed by both the new and old KSK. The new KSK is communicated to the zone owner, after which the zone owner updates the DS RRset to replace the DS record for the old KSK with a DS record for the new KSK. After the necessary DS RRset TTL period has elapsed, the old KSK can be removed from Provider A's DNSKEY RRset.
- Zone Signing Key Rollover: In Model 2, each managed-DNS provider has their own ZSK. The ZSK roll for Provider A would start with them generating a new ZSK, including it in their DNSKEY RRset, and re-signing the new DNSKEY RRset with their KSK. The new ZSK of Provider A would then be communicated to the zone owner, who would initiate the process of importing this ZSK into the DNSKEY RRsets of the other providers, using their respective APIs. Before signing zone data with the new ZSK, Provider A should wait for the DNSKEY TTL plus the time to import the ZSK into Provider B, plus the time to propagate the DNSKEY RRset to all authoritative servers of both providers. Once the necessary Pre-Publish key-rollover time periods have elapsed, Provider A and the zone owner can initiate the process of removing the old ZSK from the DNSKEY RRsets of all providers.

7. Using Combined Signing Keys

A Combined Signing Key (CSK) is one in which the same key serves the purposes of both being the secure entry point (SEP) key for the zone and signing all the zone data, including the DNSKEY RRset (i.e., there is no KSK/ZSK split).

Model 1 is not compatible with CSKs because the zone owner would then hold the sole signing key, and providers would not be able to sign their own zone data.

Model 2 can accommodate CSKs without issue. In this case, any or all of the providers could employ a CSK. The DS record in the parent zone would reference the provider's CSK instead of KSK, and the public CSK would need to be imported into the DNSKEY RRsets of all of the other providers. A CSK key rollover for such a provider would involve the following: The provider generates a new CSK, installs the new CSK into the DNSKEY RRset, and signs it with both the old and new CSKs. The new CSK is communicated to the zone owner. The zone owner exports this CSK into the other provider's DNSKEY RRsets and replaces the DS record referencing the old CSK with one referencing the new one in the parent DS RRset. Once all the zone data has been re-signed with the new CSK, the old CSK is removed from the DNSKEY RRset, and the latter is re-signed with only the new CSK. Finally, the old CSK is removed from the DNSKEY RRsets of the other providers.

8. Use of CDS and CDNSKEY

CDS and CDNSKEY records [RFC7344][RFC8078] are used to facilitate automated updates of DNSSEC secure-entry-point keys between parent and child zones. Multi-signer DNSSEC configurations can support this, too. In Model 1, CDS/CDNSKEY changes are centralized at the zone owner. However, the zone owner will still need to push down updated signed CDNS/DNSKEY RRsets to the providers via the key-management mechanism. In Model 2, the key-management mechanism needs to support cross-importation of the CDS/CDNSKEY records, so that a common view of the RRset can be constructed at each provider and is visible to the parent zone attempting to update the DS RRset.

9. Key-Management-Mechanism Requirements

Managed-DNS providers typically have their own proprietary zone configuration and data-management APIs, commonly utilizing HTTPS and Representational State Transfer (REST) interfaces. So, rather than outlining a new API for key management here, we describe the specific functions that the provider API needs to support in order to enable the multi-signer models. The zone owner is expected to use these API functions to perform key-management tasks. Other mechanisms that can partly offer these functions, if supported by the providers, include the DNS UPDATE protocol [RFC2136] and Extensible Provisioning Protocol (EPP) [RFC5731].

- The API must offer a way to query the current DNSKEY RRset of the provider.
- For Model 1, the API must offer a way to import a signed DNSKEY RRset and replace the current one at the provider. Additionally, if CDS/CDNSKEY is supported, the API must also offer a way to import a signed CDS/CDNSKEY RRset.
- For Model 2, the API must offer a way to import a DNSKEY record from an external provider into the current DNSKEY RRset. Additionally, if CDS/CDNSKEY is supported, the API must offer a mechanism to import individual CDS/CDNSKEY records from an external provider.

In Model 2, once initially bootstrapped with each other's zone-signing keys via these API mechanisms, providers could, if desired, periodically query each other's DNSKEY RRsets, authenticate their signatures, and automatically import or withdraw ZSKs in the keyset as key-rollover events happen.

10. DNS Response-Size Considerations

The multi-signer models result in larger DNSKEY RRsets, so the size of a response to a query for the DNSKEY RRset will be larger. The actual size increase depends on multiple factors: DNSKEY algorithm and keysize choices, the number of providers, whether additional keys are prepublished, how many simultaneous key rollovers are in progress, etc. Newer elliptic-curve algorithms produce keys small enough that the responses will typically be far below the common Internet-path MTU. Thus, operational concerns related to IP fragmentation or truncation and TCP

fallback are unlikely to be encountered. In any case, DNS operators need to ensure that they can emit and process large DNS UDP responses when necessary, and a future migration to alternative transports like DNS over TLS [RFC7858] or DNS over HTTPS [RFC8484] may make this topic moot.

11. IANA Considerations

This document has no IANA actions.

12. Security Considerations

The multi-signer models necessarily involve third-party providers holding the private keys that sign the zone-owner's data. Obviously, this means that the zone owner has decided to place a great deal of trust in these providers. By contrast, the more traditional model in which the zone owner runs a hidden master and uses the zone-transfer protocol with the providers is arguably more secure, because only the zone owner holds the private signing keys, and the third-party providers cannot serve bogus data without detection by validating resolvers.

The zone-key import and export APIs required by these models need to be strongly authenticated to prevent tampering of key material by malicious third parties. Many providers today offer REST/HTTPS APIs that utilize a number of client-authentication mechanisms (username/password, API keys etc) and whose HTTPS layer provides transport security and server authentication. Multifactor authentication could be used to further strengthen security. If DNS protocol mechanisms like UPDATE are being used for key insertion and deletion, they should similarly be strongly authenticated -- e.g., by employing Transaction Signatures (TSIG) [RFC2845]. Key generation and other general security-related operations should follow the guidance specified in [RFC6781].

13. References

13.1. Normative References

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/info/rfc1034>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC2845] Vixie, P., Gudmundsson, O., Eastlake 3rd, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, DOI 10.17487/RFC2845, May 2000, <https://www.rfc-editor.org/info/rfc2845>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <https://www.rfc-editor.org/info/rfc4033>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <https://www.rfc-editor.org/info/rfc4034>.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <https://www.rfc-editor.org/info/rfc4035>.

[RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, DOI 10.17487/RFC5155, March 2008, <https://www.rfc-editor.org/info/rfc5155>.

[RFC6781] Kolkman, O., Mekking, W., and R. Gieben, "DNSSEC Operational Practices, Version 2", RFC 6781, DOI 10.17487/RFC6781, December 2012, <https://www.rfc-editor.org/info/rfc6781>.

[RFC7344] Kumari, W., Gudmundsson, O., and G. Barwood, "Automating DNSSEC Delegation Trust Maintenance", RFC 7344, DOI 10.17487/RFC7344, September 2014, <https://www.rfc-editor.org/info/rfc7344>.

[RFC8078] Gudmundsson, O. and P. Wouters, "Managing DS Records from the Parent via CDS/CDNSKEY", RFC 8078, DOI 10.17487/RFC8078, March 2017, <https://www.rfc-editor.org/info/rfc8078>.

[RFC8198] Fujiwara, K., Kato, A., and W. Kumari, "Aggressive Use of DNSSEC-Validated Cache", RFC 8198, DOI 10.17487/RFC8198, July 2017, <https://www.rfc-editor.org/info/rfc8198>.

13.2. Informative References

[BLACKLIES] Valsorda, F. and Ã. GuÃ°mundsson, "Compact DNSSEC Denial of Existence or Black Lies", Work in Progress, Internet-Draft, draft-valsorda-dnsop-black-lies-00, 21 March 2016, <https://tools.ietf.org/html/draft-valsorda-dnsop-black-lies-00>.

[RFC1995] Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995, DOI 10.17487/RFC1995, August 1996, <https://www.rfc-editor.org/info/rfc1995>.

[RFC2136] Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, DOI 10.17487/RFC2136, April 1997, <https://www.rfc-editor.org/info/rfc2136>.

[RFC5731] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Domain Name Mapping", STD 69, RFC 5731, DOI 10.17487/RFC5731, August 2009, <https://www.rfc-editor.org/info/rfc5731>.

[RFC5936] Lewis, E. and A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, DOI 10.17487/RFC5936, June 2010, <https://www.rfc-editor.org/info/rfc5936>.

[RFC7129] Gieben, R. and W. Mekking, "Authenticated Denial of Existence in the DNS", RFC 7129, DOI 10.17487/RFC7129, February 2014, <https://www.rfc-editor.org/info/rfc7129>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <https://www.rfc-editor.org/info/rfc7858>.

[RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <https://www.rfc-editor.org/info/rfc8484>.

[RFC8499] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", BCP 219, RFC 8499, DOI 10.17487/RFC8499, January 2019, <https://www.rfc-editor.org/info/rfc8499>.

Acknowledgments

The initial version of this document benefited from discussions with and review from Duane Wessels. Additional helpful comments were provided by Steve Crocker, Ulrich Wisser, Tony Finch, Olafur Gudmundsson, Matthijs Mekking, Daniel Migault, and Ben Kaduk.

Authors' Addresses

Shumon Huque
Salesforce
415 Mission Street, 3rd Floor
San Francisco, CA 94105
United States of America
Email: shuque@gmail.com

Pallavi Aras
Salesforce
415 Mission Street, 3rd Floor
San Francisco, CA 94105
United States of America
Email: paras@salesforce.com

John Dickinson
Sinodun IT
Magdalen Centre
Oxford Science Park
Oxford
OX4 4GA
United Kingdom
Email: jad@sinodun.com

Jan Vcelak
NS1
55 Broad Street, 19th Floor
New York, NY 10004
United States of America
Email: jvcelak@ns1.com

David Blacka
Verisign
12061 Bluemont Way
Reston, VA 20190
United States of America
Email: davidb@verisign.com