Negotiating DNSSEC Algorithms over Legacy Proxies

Amir Herzberg\textsuperscript{1,2} and Haya Shulman\textsuperscript{2}

\textsuperscript{1} Computer Science Department, Bar Ilan University, Ramat Gan, Israel
\textsuperscript{2} Fachbereich Informatik, Technische Universität Darmstadt, Darmstadt, Germany
\{amir.herzberg,haya.shulman\}@gmail.com

Abstract. To ensure best security and efficiency, cryptographic protocols should allow parties to negotiate the use of the ‘best’ cryptographic algorithms supported by the different parties; this is usually referred to as \textit{cipher-suite negotiation}, and considered an essential feature of such protocols, e.g., TLS and IPsec. However, such negotiation is absent from protocols designed for \textit{distribution} of cryptographically-signed objects, such as DNSSEC. One reason may be the challenges of securing the choice of the ‘best’ algorithm, especially in the presence of intermediate ‘proxies’ (crucial for performance), and in particular, providing solutions, compatible with the existing legacy servers and proxies; another reason may be a lack of understanding of the security and performance damages due to lack of negotiation.

We show that most DNSSEC signed domains, support only RSA 1024-bit signatures, which are considered insecure, and are also larger than alternatives; the likely reason is lack of negotiation mechanisms. We present a DNSSEC-negotiation mechanism, allowing name-servers to send responses containing only the keys and signatures required by the requesting resolver. Our design is compatible with intermediary proxies, and even with legacy proxies, that do not support our negotiation mechanism. We show that our design enables incremental deployment and will have negligible performance impact on overhead of DNSSEC as currently deployed, and significant improved performance to DNSSEC if more domains support multiple algorithms; we also show significant security benefits from the use of our design, under realistic, rational adoption model. Ideas of our design apply to other systems requiring secure and efficient distribution of signed data, such as wireless sensor networks (WSNs).

1 Introduction

A \textit{cipher-suite} is an ordered set of (one or more) cryptographic algorithms, each implementing a corresponding function among the functions used by a cryptographic protocol. For example, the \texttt{RSA\_WITH\_RC4\_128\_MD5} cipher suite uses RSA for key exchange, RC4 with a 128-bit key for bulk encryption, and MD5 for message authentication. \textit{Cipher-suite negotiation} refers to the process of selecting
the cipher-suite to be used in a protocol between two (or more) parties, among multiple cipher-suites supported by each of the participants. Many standard cryptographic protocols, e.g., IKE, SSH, SSL and TLS, [RFC2409, RFC4253, RFC6101, RFC5246] use cipher-suite negotiation to ensure that the parties select the ‘best’ cipher-suite among those they jointly support, in order to avoid broken algorithms and to facilitate migration to better (more secure, more efficient) algorithms.

Currently, DNSSEC is an exception: it allows the use of multiple signature algorithms and hash functions, e.g., RSA and elliptic curves (see [7] for a complete list of cryptographic algorithms). However, no mechanism allows name servers to identify the best set of algorithms, keys and signatures to send in response to a particular request, i.e., for cipher-suite negotiation. As a result, during a DNS transaction between a resolver and a name server, all the keys and signatures, supported by the target zone, are sent to the resolver, even if some of those algorithms are unsupported or unvalidated by the resolver.

We collected responses’ sizes from Top Level Domains (TLDs) and Alexa-top-million domains, [1]; our measurements are plotted in Figure 1. The measurements show that the overhead of signed DNS responses is significant in comparison to plain DNS responses. For instance, non-existent domain (NXD) is a very common response, which often occurs due to a typo in a DNS query: the size of NXD responses without DNSSEC is less than 400 bytes, while with DNSSEC, 70% of the responses exceed 1000 bytes and 10% are even larger than the link’s Maximal Transmission Unit (MTU) (which also holds for more than 30% of DNSKEY responses). Signed responses for ANY query type can reach even 5000 bytes and more, while plain ANY type responses are less than 1000 bytes. Large DNSSEC signed responses inflict significant overhead on the network and on the end points to the DNS transation, and often result in failures; we describe the problems in Section 3. We believe that the problems with large responses, and the lack of cipher-suite negotiation mechanism, motivate administrators to use only a limited number of cipher-suites. In particular, without cipher-suite negotiation mechanism administrators are likely to avoid algorithms with larger keys/signatures, e.g., 2048 or 4096 bit RSA; these may offer better security, but surely will increase response length significantly and as a result also exacerbate the interoperability problems and exposure to attacks.

![Fig. 1. Length of responses for signed and non-signed Alexa and TLDs, for ANY, DNSKEY and A resource records; A records were sent for random subdomains of tested domains, and resulted in NXD responses.](image-url)