POSTER: Predictive Cipher-Suite Negotiation For Boosting Deployment of New Ciphers

Elias Heftrig
ATHENE
Fraunhofer SIT
Germany

Jean-Pierre Seifert
Technische Universität Berlin
Fraunhofer SIT
Germany

Haya Shulman
ATHENE
Fraunhofer SIT
Germany

Michael Waidner
ATHENE
Technische Universität Darmstadt
Fraunhofer SIT
Germany

Nils Wisiol
Technische Universität Berlin
Germany

ABSTRACT

Deployment of strong cryptographic ciphers for DNSSEC is essential for long term security of DNS. Unfortunately, due to the hurdles involved in adoption of new ciphers coupled with the limping deployment of DNSSEC, most domains use the weak RSA-1024 cipher.

The main problem towards deployment of new ciphers is the resulting bloat of DNSSEC-signed responses due to support of multiple ciphers. This causes not only load on network, but worse, it results in DNS lookup failures, e.g., many network devices block such huge packets. Merely dropping the old ciphers and moving to use new stronger ciphers is not an option since this would break the DNS functionality for all the clients which do not support those new ciphers. The requirement to support new ciphers on both clients and servers coupled with the possible DNS failures due to the resulting large responses reduces the motivation to improve the security of DNS.

We aim to resolve this vicious circle. In this work we propose an approach for deployment of new ciphers using a single-sided cipher-suite negotiation. Our mechanism uses machine learning for inferring the set of ciphers potentially supported by the client and then selecting the best cipher from that list. Our evaluations demonstrate that our single-sided cipher-suite negotiation not only allows the domains to unilaterally improve security without waiting for clients to integrate support for new secure ciphers, but it also improves DNS performance by reducing failures.

Our results show that a single sided solution can, not only push adoption of new ciphers forward, but it also will resolve the existing interoperability problems with DNSSEC. Our design and preliminary analysis on the feasibility of applying machine learning to this context results in more secure and available DNSSEC. We outline our methodology for machine learning assisted cipher-suite negotiation and provide steps and challenges for future research.

CCS '21, November 15–19, 2021, Virtual Event, Republic of Korea

ACM ISBN 978-1-4503-8454-4/21/11.
https://doi.org/10.1145/3460120.3485349

CCS CONCEPTS

• **Security and privacy** → **Network security**; **Cryptography**; • **Computing methodologies** → **Machine learning**.

KEYWORDS

DNSSEC, Cipher-Suite Negotiation, Crypto-agility, Machine Learning

ACM Reference Format:
Elias Heftrig, Jean-Pierre Seifert, Haya Shulman, Michael Waidner, and Nils Wisiol. 2021. POSTER: Predictive Cipher-Suite Negotiation For Boosting Deployment of New Ciphers. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS '21), November 15–19, 2021, Virtual Event, Republic of Korea.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3460120.3485349

1 INTRODUCTION

DNSSEC [RFC4033-RFC4035] was proposed more than two decades ago, yet is still not widely deployed, only about 1% of the domains are signed and most DNS resolvers do not validate responses. Worse, even the signed domains often still remain vulnerable due to misconfigured or weak cryptographic keys, [1, 5]. Unfortunately, supporting multiple ciphers does not solve the problem since the additional cryptographic material of DNSSEC increases the packet size significantly causing the signed DNS responses to be blocked by the intermediate network devices. On the other hand, selecting and using just one strong cipher is also not an option since many DNS resolvers do not have support for new ciphers. When a resolver receives records signed with ciphers it does not understand, the validation fails. As a result, a large fraction of the DNS servers still support RSA 1024, [4]. Even now, when most resolvers support old ciphers and insufficiently strong keys, DNSSEC-signed responses are often blocked causing the clients connectivity problems. Proposals for cipher suite negotiation of DNSSEC, [2, 3], did not see adoption since they require modifications to both the resolvers and the nameservers. Other proposals consider out-of-band channels for signalling the supported ciphers such as in transport layer EDNS mechanism [RFC6975]. Although standardised eight years ago they are still not supported by the DNS resolvers in the Internet.

Our contribution. In this work, we propose a new approach for integrating ciphers into DNSSEC. According to our proposal the domains can unilaterally deploy new ciphers. The key idea

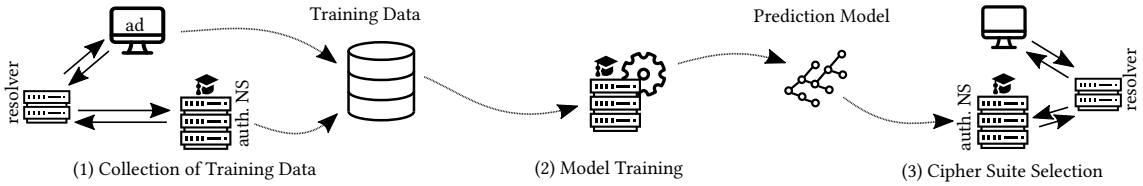

Figure 1: Visualization of the learning process. (1) Training data is collected by deploying an ad network. The ad on a client triggers queries to our nameserver. We collect information about the query success from within the ad client and information about the resolver's query from the queries at our authoritative name server. (2) The collected data is used to train a predictive model using machine learning. (3) The model is used to predict the supported ciphers in a real-world scenario.

behind our proposal is that the nameservers can extrapolate based on the information learnt from the DNS requests which ciphers the resolvers probably cannot handle. On the nameserver side we continually monitor the communication with the resolvers, and dynamically build models using Machine Learning (ML) about the supported ciphers by each resolver. The models are built based on caching and query behaviour, as well as properties in the IP and transport headers of the DNS requests. For learning the behaviour of the resolvers we use decision tree (DT) based approach. From the set of predicted supported ciphers, the cipher for the DNS response is chosen based on server-side preference. The classifiers that predict algorithm support are pre-trained based on a large volume of network traces with queries and responses to open resolvers on the Internet and popular resolver software in our lab. Next, our server-side module selects the best (i.e., the strongest) cipher which is supported by the resolver, among the ciphers used by the domain.

Our mechanism allows the domains to unilaterally adopt new ciphers without exposing the clients to potential connectivity failures. The clients with DNS resolvers that support the new ciphers immediately enjoy the additional security benefits. On the other hand, our mechanism reduces the failures when the nameservers communicate with resolvers that support multiple ciphers or ciphers with multiple keys of different sizes.

2 SERVER-ONLY CIPHERS NEGOTIATION

Design. The goal is to identify the best cipher for each given resolver and to send responses using only that cipher, On the nameserver we create n copies of the zonefile for each of the n ciphers supported by a given domain. Each copy of the zonefile is signed by one cipher. During the communication with the resolvers the server applies ML to decide on the optimal cipher to use and provides records from the corresponding zonefile.

ML-Features. The server uses the queries from the DNS resolvers to decide on the most suitable cipher for communication with that resolver. We extract features from the time domain, from TCP/IP headers and the payload of the DNS requests packets sent by the resolver.

• Time domain: these features are related to the query behaviour of DNS resolvers.
• Transport layer features: these are features in EDNS and the transport layer protocol that is used for communication.
• DNS header features: these are parameters in DNS header.
• IP header features: these are parameters which can be used to fingerprint the operating system of the DNS resolver.

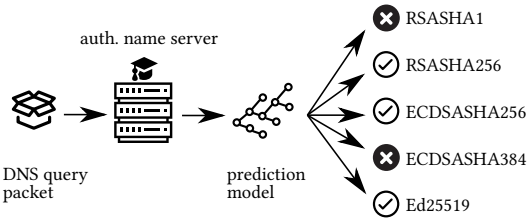

Figure 2: Schematic view of how the cipher suit(s) are selected using models trained with machine learning. The decision is based on the features that we defined. In the shown case, a response with `Ed25519` signatures is sent.

• Payload: the DNS queries and the caching behaviour of the resolver.

This information is then fed into a pre-trained model that will determine a prediction for the set of cipher suites understood by the resolver in question. The response given by the authoritative name server will then use the best of the predicted cipher suites. The process is illustrated in Figure 2.

Ad Client Data			Server Data			
Client	Query Name	Result	Remote	IP TTL	TC bit	...
`0xab`	`ab.rsasha256...`	ok	`2a::42`	3	set	...
`0xab`	`ab.ecdsasha256...`	ok	`2a::42`	4	no	
`0xab`	`ab.ed25519...`	fail	`2a::42`	4	set	
`0xc0`	`c0.rsasha256....`	ok	`1.1.1.1`	3	...	

Table 1: Training data.

Model training. The models used for cipher selection are trained using machine learning techniques. We describe the training process in Figure 1. For any training, it is essential to obtain feedback as to whether validation using the selected cipher suite was successful. We collect this data using an ad network, where a client computer is instructed by the ad to query certain zones in the DNS using its pre-configured resolver. Whether or not validation happened, and whether or not the validation was successful, can then be determined for the resolver in use. This is done for a number of different cipher suits[1]. By encoding a unique client-id in the DNS query, this data is matched up with the DNS traffic data that is recorded at the authoritative name servers of our domain, to create training sets for the prediction models; An example is displayed in Table 1.

[1]Observe that we implicitly also testing whether or not the cipher suites used by any parent zones are supported. Of course, we expect very broad support for the cipher suites used by root and top level zones.

In our measurements we collected a total of 207 resolvers covering our 8 lab resolvers, 13 popular open resolver services and 186 randomly selected validating resolvers from UDP/53 port scans resolve our domains. Using the RCODE and AD flag from the query response at the client under our control, we were able to determine, which resolvers support which algorithms and with what key sizes (RSA only). The results are shown in Table 2.

Algorithm	KeyLength	Share of Resolvers
ecdsap256sha256	256	95.16%
ecdsap384sha384	384	74.19%
ed25519	256	78.49%
ed448	456	13.98%
rsasha1	1024	37.63%
	1871	37.63%
	2048	37.63%
	4096	37.10%
rsasha1nsec3sha1	1024	74.73%
	1871	75.27%
	2048	74.73%
	4096	74.19%
rsasha256	1024	75.27%
	1871	76.34%
	2048	75.27%
	4096	72.58%
rsasha512	1024	75.27%
	1871	76.34%
	2048	74.73%
	4096	73.12%

Table 2: Resolvers and the supported ciphers and key lengths.

Once the training data is gathered, it must be decided which kind of model we use. In this work we train a decision tree, as we expect decision processes to not be overly complicated. Furthermore, such models can be nicely analyzed and explained.

Implementation. We develop our mechanism as a separate module which implements our ML classifier and based on the requests that it receives from the resolver, identifies the optimal cipher to use. We also develop a script which given a zonefile on the nameserver, and the cryptographic keys, generates a separate zonefile per cipher each signed with the corresponding key. Once the optimal cipher is identified our module signals to the script on the nameserver which zonefile copy should be used (protected with the corresponding optimal cipher).

3 EVALUATION

In this section we present our preliminary evaluation results. After obtaining the classifiers for each cipher, we checked the accuracy of their prediction by comparing to the test set (the part of the collected data which was not used for training). Each classifier shows moderate accuracy, with the proportion of false predictions making up about one third of test examples. The detailed metrics are shown in Table 3. An increase in training data or an increase in the number of features would further improve the individual classifier quality. Most importantly, our preliminary work, including the classifier and the training data, is currently based on a single DNS query to the server. However, in typical scenarios, the resolving process involves several queries to the server, which further improve precision. Notice however, that the individual classifier's wrong guesses only mean that our mechanism predicted another better cipher, which is still correctly supported by the resolver.

To assess the performance of our cipher selection mechanism, we used the classifiers to predict which cipher would be selected for the DNS queries contained in our test set. Since for each query in the test set the responsible DNS resolver, that sent it, is known, the correctness of the prediction can be assessed using the resolver cipher support that we collected during our evaluation of the resolvers in the lab.

algorithm	true neg.	true pos.	false neg.	false pos.
rsasha1	80%	0%	20%	0%
rsasha1nsec3sha1	39%	25%	22%	14%
rsasha256	39%	26%	22%	13%
rsasha512	35%	26%	21%	18%
ecdsap256sha256	30%	20%	31%	18%
ecdsap384sha384	39%	21%	24%	16%
ed25519	43%	16%	28%	13%
ed448	86%	4%	5%	5%

Table 3: Performance of the decision-tree classifiers.

The results of this procedure show that the choice of algorithm in more than 99% of the cases matches the best cipher supported by the involved resolver. In cases where the prediction was incorrect, the resolver's capabilities were mostly underestimated, resulting in weaker security than what could be possible with the resolver in question. On the other hand, cases where the resolver's capabilities were overestimated, and validation at the resolver side was skipped, are relatively few. Overall, for almost all resolvers, the security that could be achieved with the cipher chosen by our mechanism is higher than what would be achieved with setting the cipher fixed to the recommended setting, i.e., ecdsap256sha256. (The exceptions being bind9113, kresd532, and cznic-odvr, with the latter presumably using kresd).

4 CONCLUSIONS

We propose a new approach for deployment of ciphers in DNSSEC, which eliminates the requirement to change both ends of the communication. We use ML to learn the client side behaviour and identify an optimal cipher that the server can use which is also supported by the DNS resolver. Our initial evaluations show 99% success on our dataset of resolvers in the Internet. Our approach not only pushes deployment of new ciphers forward, but also reduces the failures with DNSSEC signed responses.

ACKNOWLEDGEMENTS

This research has been funded by the German Federal Ministry of Education and Research and the Hessen State Ministry for Higher Education, Research and Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

REFERENCES

[1] Tianxiang Dai, Haya Shulman, and Michael Waidner. 2016. Dnssec misconfigurations in popular domains. In *International Conference on Cryptology and Network Security*. Springer, 651–660.

[2] Amir Herzberg and Haya Shulman. 2015. Cipher-Suite Negotiation for DNSSEC: Hop-by-Hop or End-to-End? *IEEE Internet Computing* 19, 1 (2015), 80–84.

[3] Amir Herzberg, Haya Shulman, and Bruno Crispo. 2014. Less is more: cipher-suite negotiation for DNSSEC. In *Proceedings of the 30th Annual Computer Security Applications Conference*. 346–355.

[4] Moritz Müller, Willem Toorop, Taejoong Chung, Jelte Jansen, and Roland van Rijswijk-Deij. 2020. The Reality of Algorithm Agility: Studying the DNSSEC Algorithm Life-Cycle. In *Proceedings of the ACM Internet Measurement Conference*. 295–308.

[5] Haya Shulman and Michael Waidner. 2017. One Key to Sign Them All Considered Vulnerable: Evaluation of {DNSSEC} in the Internet. In *14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17)*. 131–144.