Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 20, 2019

D. Kaiser
University of Luxembourg

November 16, 2018

Efficient Hinting for Privacy Preserving DNS-SD using Bloomfilters
draft-kaiser-dnssd-bloomfilter-hints-00

Abstract

While DNS-SD over mDNS significantly improves the convenience of network configuration, parts of the published information may seriously breach the users' privacy. Currently discussed privacy extensions either are not efficient in terms of multicast messages sent, reduce privacy and complicate key revocation by introducing an 1:m pairing system, or use trial encryptions which are inefficient in terms of necessary computational power.

The method proposed in this document leverages Bloomfilters to significantly reduce the number of multicast (public) messages for a DNS-SD privacy extension based on an 1:1 pairing mechanism. This allows keeping the advantages of both an 1:1 pairing system and a hinting system that does not require trial encryptions, while mitigating the main disadvantage: multicast messages sent.

Table of Contents

1. Introduction

DNS-SD [RFC6763] over mDNS [RFC6762] enables zero-configuration service discovery in local networks. While it significantly improves the convenience of network configuration, parts of the published information may seriously breach the users' privacy. These privacy issues and potential solutions are discussed in [KW14a], [KW14b], and [K17].

[[TODO]]

This document proposes leveraging Bloomfilters to significantly reduce the number of multicast (public) messages for a DNS-SD privacy extension like [I-D.ietf-dnssd-privacy], which is based on an 1:1 pairing mechanism (e.g. [I-D.ietf-dnssd-pairing]).

1.1. Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Bloomfilter-based Discovery Protocol

2.1. Basic Idea

Instead of transmitting a lot of discovery messages containing HASH(<nonce>|<pairing key>), sending a single discovery message containing a Bloomfilter over the respective hashes will significantly reduce the number of necessary discovery messages.

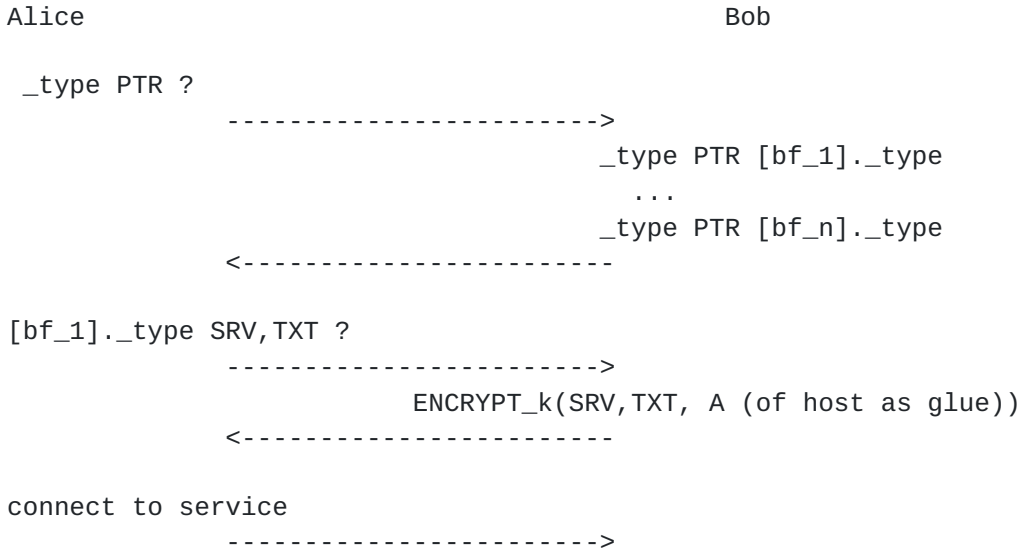
False positives are not a problem. They will only cause an additional pair of unicast messages.

2.2. Overview

This section provides an overview over Bloomfilter-based hinting, illustrated by various scenarios where Alice searches for service instances of type _type and Bob offers such an instance. This type could be a _psds service instance for a two-stage discovery system, or any other type for a one-stage discovery system.

In the following, [bf_1],...,[bf_n] are Bloomfilters whose construction is described in Section 3.2. As we can store at least 25 hints in one Bloomfilter with a very low false positive rate (see Section 3.1), n is expected to be very low.

If a pairing exists:

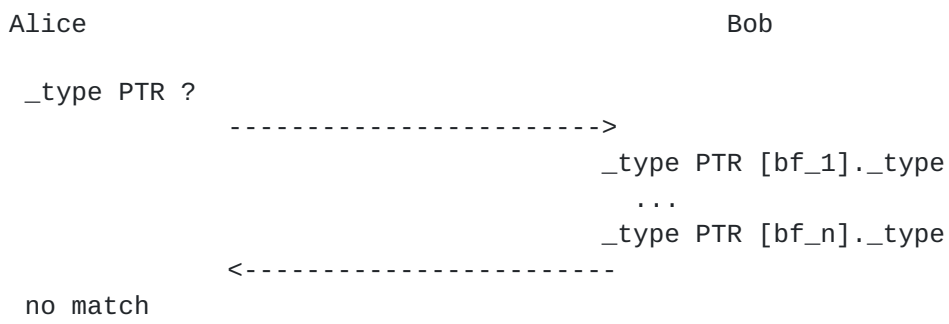


Only the first two messages are multicast (public).

The encrypted message SHOULD be padded in such a way that each answer message has the same length, so that answers from the server are indistinguishable from randomly selected bits for an unpaired device.

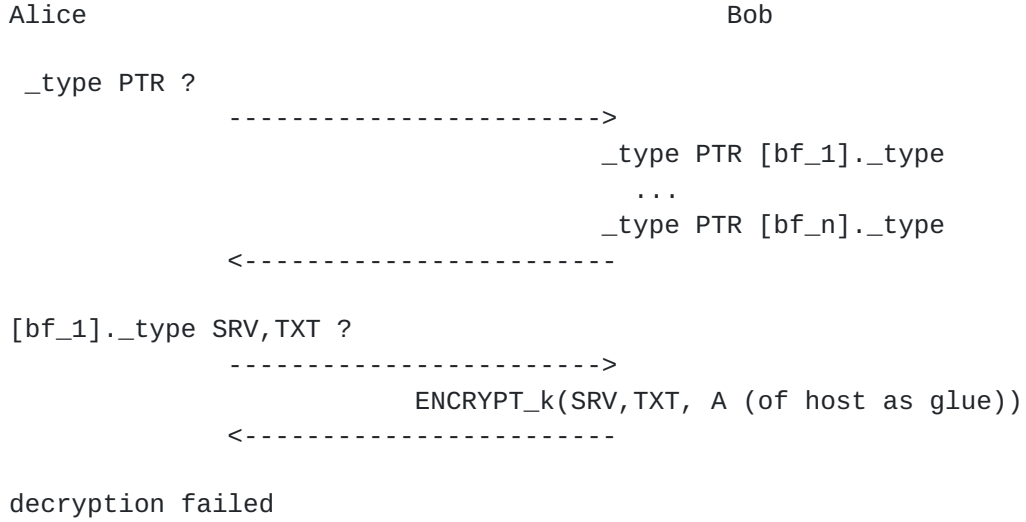
For checking a hint, Alice pre-calculates a list of HASH(derive(secret)||nonce) for all her pairings per time interval, and checks if any of these are in the Bloomfilter. This is even more efficient than checking whether n received hashes are in a pre-calculated hash table as described in [I-D.ietf-dnssd-privacy].

If no pairing exists, and the hint is not false positive:



In this case, a lot of messages are saved, as a severely compressed version (1:25) of the hints was sufficient for Alice to realize that this service instance was not meant for her.

If no pairing exists, and the hint is a false positive:



In the case of a false positive, only a pair of additional multicast messages and the corresponding cryptographic operations are needed. With a false positive rate of 1:16000 (see Section 3.1), this effect is negligible.

This case also applies to an attacker trying to deceive Bob.

2.3. Direct Resolving

[[TODO: Show a diagram of the message flow for direct resolving.]]

3. Bloomfilter Hints

3.1. Performance Analysis

As specified in [RFC6763], the maximum length of a service instance name is 63 bytes. As DNS labels are allowed to contain binary data, this allows a 504 bit wide Bloomfilter.

Using classical Bloomfilters [[we could discuss more efficient alternatives]] setting the maximum hints per Bloomfilter to 25 results in a desirable false positive rate of 1 in 16000. This means, using the proposed Bloomfilter-based hinting method, the necessary multicast (public) discovery messages can be reduced by factor 25 at the cost of one additional set of messages for every 16000 discovery messages. Further, the server needs additional computational power for constructing the bloomfilter. However, given the efficiently of Bloomfilter construction, this is negligible. The difference in needed computational power on the client is negligible as well.

[[TODO: elaborate]]

3.2. Construction of Bloomfilter Hints

The Bloomfilters, [bf_1],...,[bf_n], in the protocol description above, are constructed as follows:

- Initialize bf_1 as a 504 bit wide Bloomfilter.
- For each paired client p, put an identifier of the form HASH(derive(secret_p)||nonce) into a Bloomfilter bf_1. The nonce is constructed as described in Section 3.4 of [I-D.ietf-dnssd-privacy].
- If there are 25 elements in the Bloomfilter, start a new Bloomfilter bf_{i+1} and repeat from step 2.
- Use the Bloomfilters bf_1,...,bf_n as service instance names of service instances of type _type.

4. Security Considerations

[[TODO]]

5. IANA Considerations

This draft does not require any IANA action.

6. Acknowledgments

7. Informative References

[I-D.ietf-dnssd-pairing]
Huitema, C. and D. Kaiser, "Device Pairing Using Short Authentication Strings", draft-ietf-dnssd-pairing-05 (work in progress), October 2018.

[I-D.ietf-dnssd-privacy]
Huitema, C. and D. Kaiser, "Privacy Extensions for DNS-SD", draft-ietf-dnssd-privacy-05 (work in progress), October 2018.

[K17] Kaiser, D., "Efficient Privacy-Preserving Configurationless Service Discovery Supporting Multi-Link Networks", 2017, <http://nbn-resolving.de/urn:nbn:de:bsz:352-0-422757>.

[KW14a] Kaiser, D. and M. Waldvogel, "Adding Privacy to Multicast DNS Service Discovery", DOI 10.1109/TrustCom.2014.107, 2014, <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7011331>.

[KW14b] Kaiser, D. and M. Waldvogel, "Efficient Privacy Preserving Multicast DNS Service Discovery", DOI 10.1109/HPCC.2014.141, 2014, <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7056899>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <https://www.rfc-editor.org/info/rfc6762>.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <https://www.rfc-editor.org/info/rfc6763>.

Author's Address

Daniel Kaiser
University of Luxembourg
6, avenue de la Fonte
Esch-sur-Alzette 4364
Luxembourg

Email: daniel.kaiser@uni.lu
URI: https://secan-lab.uni.lu/