CORE
Internet-Draft
Intended status: Standards Track
Expires: September 1, 2016

T. Fossati
Nokia
H. Tschofenig
ARM Ltd.

February 29, 2016

Introducing Server Name Identifiers in Certificates
draft-fossati-core-server-name-id-00.txt

Abstract

TBD.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 1, 2016.

Table of Contents

1. Introduction

Today, many Internet of Things (IoT) deployments consist of an IoT device that interacts with a cloud service infrastructure. (This deployment model is described in Section 2.2 of [RFC7452].) If TLS/DTLS is used to mutually authenticate the device and the cloud server, then the guidance in [I-D.ietf-dice-profile] - which, in turn, takes [RFC7252] recommendations into account - should be followed.

Let us take the CoAP protocol as an example. According to Section 9.1.3.3 of [RFC7252], a DTLS client that receives a certificate from the DTLS server must check that the authority of the requested URI matches "at least one of the authorities of any CoAP URI found in a field of URI type in the SubjectAltName (SAN) set. If there is no SubjectAltName in the certificate, then the authority of the request URI must match the Common Name (CN) found in the certificate [...].". A URI that includes an authority, such as a 'coaps' URI, needs to include a fully qualified domain name (FQDN), or an IP literal as its host part, as stated in Section 4.2.1.6 of [RFC5280]. So, an IoT device that wants to talk to a CoAP server at coaps://example.com will expect to receive a certificate with a matching URI in either the content of the SAN extension or the CN.

The Server Name Indication (SNI) extension [RFC6066] defined for TLS/DTLS allows a client to tell a server the name of the server it is contacting. This is a feature useful when the server is part of a hosting solution where multiple virtual servers are using a single underlying network address. Section 3 of [RFC6066] only allows FQDN hostname of the server in the ServerName field.

When a TLS/DTLS server has an FQDN registered in the DNS then the use of certificates work well with TLS/DTLS to secure protocols like HTTP or CoAP. While the DNS can be taken for granted in the Web, many IoT deployments do not mandate its presence. There are even IoT deployments where the server infrastructure is located in a residential environment in which IoT devices interact with the server solely in the local network (see also Section 2.1 of [RFC7452]).

Since static configuration is not generally a viable option from a usability point of view, in order to cope with scenarios like the one described above there is a need to define some kind of stable, non-DNS-based identifier that can be used with certificates. Note that an alternative is to avoid using certificates altogether and to instead use raw public keys. With raw public keys, the raw public key itself is the identifier and some out-of-band validation technique is needed instead, as described in RFC 7250 [RFC7250].

This document specifies such identifiers for use with certificates.

2. Terminology and Requirements Language

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].

3. Syntax

[Editor’s Note: This is a strawman proposal for the identifier definition.]

This section defines the syntax for the instance and the domain component, which are separated by the "@" sign. The structure for the name and the domain component are determined by the namespace prefix. We call this new construct the ’Server Name Identifier’ (SN-ID).

The following ABNF reuses ALPHA and DIGIT from [RFC5234].

```
char = ALPHA / DIGIT / "-" / "_" / "~" / "!" /
          "$" / "&" / "’" / "(" / ")" / "*" /
          "," / ";" / "="
ns = ALPHA *(ALPHA / DIGIT / "-")
name = 1*RD-char
domain = 1*63RD-char
authority = [ ns "." ] name [ "@" domain ]
```

Note that the "@" and the "." signs are illegal characters in the name and the ns components.

Here are some examples:

- eui64.01-23-45-67-89-ab-cd-ef
- imei.+123456789012345
- uuid.84c05e54-1d3c-48b6-bf12-c11c55f8fdac@foo

4. Server Name Indication (SNI) Name Type and Server Name Syntax

In order to encode the SN-ID in a ServerNameList, the extension_data field of the server_name extension is expanded to allow the SN-ID in a ServerName extension:

```
struct {
    NameType name_type;
    select (name_type) {
        case host_name: HostName;
        case sn_id: AuthorityType;
    } name;
} ServerName;

enum {
    host_name(0),
    sn_id(1),
    (255)
} NameType;

opaque AuthorityType<1..2^16-1>;
```

AuthorityType, the data structure associated with the authority declaration, is a variable-length vector that begins with a 16-bit length field indicating the length of the following authority. The value in the authority field is represented as a byte string using ASCII encoding. It MUST NOT contain any percent-encoded character other than for those characters not explicitly allowed by the grammar in Section 3.

5. Subject Alternative Name Extension

As described in RFC 5280 [RFC5280] the subjectAltName may carry additional name types through the use of the otherName field. The format and semantics of the name are indicated through the OBJECT IDENTIFIER in the type-id field. The name itself is conveyed as value field in otherName. This document defines a new value for the type-id field.

This section defines the SN-ID as a form of otherName from the GeneralName structure in SubjectAltName defined in [RFC5280].

```
id-sn-id OBJECT IDENTIFIER ::= { id-pkix id-sn-id(TODO) }
```

An X.509 server certificate intended to be used with this specification MUST contain an otherName SAN identified using a type-id of 'id-sn-id-san'.

```
id-sn-id-san OBJECT IDENTIFIER ::= { id-sn-id 2 }
```

The value field of the otherName MUST contain the SN-ID, as described in Section 3, encoded as a IA5String.

6. Client Behaviour

TLS/DTLS clients behave as follows:

1) Clients MAY include an extension of type "server_name" in the (extended) client hello. A client supporting this specification MAY include one (and one only) ServerName element conveying the SN-ID.

2) Process the Certificate message, verify the digitial signature and perform path validation (as described in Section 3.2 of RFC 5280).

3) Verify that the intended server name is indeed one of the identities bound to the presented certificate, by checking that the name in the SAN otherName of type id-sn-id-san matches the authority requested via server_name.

4) Upon receiving the CertificateRequest message, send the certificate via a Certificate message - or CertificateURL message, if the client_certificate_url extension has been successfully negotiated during the "hello" exchange.

5) Send ClientKeyExchange and then CertificateVerify to complete the mutual authentication process.

7. Server Behaviour

TLS/DTLS servers behave as follows:

1) A server receiving the extended ClientHello carrying a server_name extension uses the given server_name (with the included SN-ID) to select the appropriate certificate. The selected certificate MUST include a SAN otherName with an id-sn-

id-san type-id and value, which MUST match the requested ServerName;

a) If no certificate can be selected, the server MUST terminate the handshake by sending a fatal-level unrecognized_name(112) alert. [[CREF1: Prefer a single, hard failure, path over soft failure, or worse: ignoring the error altogether. Rationale: do not waste time/energy; provide clear and prompt diagnostic to the peer. It doesn't look like the condition that could be exploited by a timing attack.]]

b) If a matching certificate exist, the server SHALL include an extension of type "server_name" in the (extended) ServerHello message with an empty value.

2) The server MUST send the selected certificate to the client in the Certificate message.

3) Server MAY request a client certificate via a CertificateRequest message and conclude its negotiation with a ServerHelloDone message.

4) When server receives the Certificate message from the client it MUST process the Certificate message, verify the digitial signature of the certificate and perform path validation (as described in Section 3.2 of RFC 5280).

a) If the client certificate processing fails then the server MUST tear down the exchange.

b) If the client certificate processing is successful then the server finalizes the TLS handshake.

5) The server application running on top of the TLS/DTLS stack MUST check the included client identity against the access control policy at the server. It is important to note that this verification check is done outside the TLS/DTLS stack; failure to do it at the application layer may result in unauthorized access.

8. Example

In this section we discuss a more complete scenario where the mechanism described in this document is practical. Consider the following setup where IoT devices are located in a small home network with a Resource Directory (RD) [I-D.ietf-core-resource-directory] helping with discovery.

A resource directory is an entity that acts as a centralized store where protocol endpoints can register their resources and thereby make them available to others. Other devices subsequently use the resource directory to lookup devices and their resources.

The RD defines the concept of an "endpoint name" which identifies a given endpoint within a "domain". Endpoint (EP) is a term used to describe a web server or client in [RFC7252]. In the context of [I-D.ietf-core-resource-directory] an endpoint is used to describe a web server that registers resources to the resource directory. An endpoint is identified by its endpoint name, which is included during registration, and is unique within the associated domain of the registration.

Imagine various IoT devices registering their resources with the pre-configured RD (or dynamically discoverd RD). Section 5.2 of [I-D.ietf-core-resource-directory] contains a description of registration procedure using CoAP and offers examples. The resource server stores, among other things, the endpoint name and (optionally) a domain.

Once the resources are registered nodes may use the resource directory to discover the resources offered by others. Section 7 of [I-D.ietf-core-resource-directory] describes the discovery procedure and lists examples. A node may, for example, search for resources of type ’temperature’ and learns the network addresses of the nodes hosting those resources as well as their endpoint name (and, if available, their domain).

Once the network address has been obtained, direct communication between the two entities can be initiated. During the subsequent DTLS exchange to secure CoAP the server hosting the resources offers his certificates and the client executes the steps outlined in Section 6 to match the endpoint name (and optionally the domain) learned through the resource directory with the SN-ID provided in the server certificate. Note that it is not envisioned that the client compares the input to the discovery procedure with the SN-ID. In this example the input to the discovery procedure with the resource directory was the resource type, i.e., the ’temperature’ string. It is therefore assumed that the client trusts the resource directory to return genuine mappings from abstract search terms to specific servers hosting those resources.

9. IANA Considerations

TBD: This document requires registration of various identifiers into existing registries,namely

- id-sn-id
- OtherName.type-id::id-sn-id-san
- NameType::sn-id
- ServerName.name::Authority

10. Security Considerations

It’s the responsibility of the CA issuing the certificate to verify the content of the certificate before issuing a new certificate. In particular, the CA MUST ensure uniqueness of the issued certificates and that the included SN-ID is indeed correct. This should exclude the threat of a (possibly rogue) node to successfully impersonate another node’s identity.

Security considerations from Section 11.1 of [RFC6066] fully apply.

11. Acknowledgements

We would like to thank Martin Thomson, Carsten Bormann, Andrew McGregor, and Zach Shelby for their feedback during IETF 92.

12. References

12.1. Normative References

[I-D.ietf-core-resource-directory]
Shelby, Z. and C. Bormann, "CoRE Resource Directory", draft-ietf-core-resource-directory-02 (work in progress), November 2014.

[I-D.ietf-dice-profile]
Tschofenig, H. and T. Fossati, "TLS/DTLS Profiles for the Internet of Things", draft-ietf-dice-profile-17 (work in progress), October 2015.

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

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <http://www.rfc-editor.org/info/rfc5280>.

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <http://www.rfc-editor.org/info/rfc6066>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <http://www.rfc-editor.org/info/rfc7252>.

12.2. Informative References

[RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <http://www.rfc-editor.org/info/rfc7250>.

[RFC7452] Tschofenig, H., Arkko, J., Thaler, D., and D. McPherson, "Architectural Considerations in Smart Object Networking", RFC 7452, DOI 10.17487/RFC7452, March 2015, <http://www.rfc-editor.org/info/rfc7452>.

Authors' Addresses

Thomas Fossati
Nokia
3 Ely Road
Milton, Cambridge CB24 6DD
Great Britain

Email: thomas.fossati@nokia.com

Hannes Tschofenig
ARM Ltd.
110 Fulbourn Rd
Cambridge CB1 9NJ
Great Britain

Email: Hannes.tschofenig@gmx.net
URI: http://www.tschofenig.priv.at