

Workgroup: Network Working Group
Internet-Draft: draft-ietf-uta-rfc6125bis-11
Obsoletes: [6125](#) (if approved)
Published: 2 March 2023
Intended Status: Standards Track
Expires: 3 September 2023
Authors: P. Saint-Andre R. Salz
 independent Akamai Technologies
 Service Identity in TLS

Abstract

Many application technologies enable secure communication between two entities by means of Transport Layer Security (TLS) with Internet Public Key Infrastructure Using X.509 (PKIX) certificates. This document specifies procedures for representing and verifying the identity of application services in such interactions.

This document obsoletes RFC 6125.

Discussion Venues

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the Using TLS in Applications Working Group mailing list (uta@ietf.org), which is archived at <https://mailarchive.ietf.org/arch/browse/uta/>.

Source for this draft and an issue tracker can be found at <https://github.com/richsalz/draft-ietf-uta-rfc6125bis>.

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 <https://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 3 September 2023.

Copyright Notice

Copyright (c) 2023 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

- [1. Introduction](#)
 - [1.1. Motivation](#)
 - [1.2. Applicability](#)
 - [1.3. Overview of Recommendations](#)
 - [1.4. Scope](#)
 - [1.4.1. In Scope](#)
 - [1.4.2. Out of Scope](#)
 - [1.5. Terminology](#)
- [2. Identifying Application Services](#)
- [3. Designing Application Protocols](#)
- [4. Representing Server Identity](#)
 - [4.1. Rules](#)
 - [4.2. Examples](#)
- [5. Requesting Server Certificates](#)
- [6. Verifying Service Identity](#)
 - [6.1. Constructing a List of Reference Identifiers](#)
 - [6.1.1. Rules](#)
 - [6.1.2. Examples](#)
 - [6.2. Preparing to Seek a Match](#)
 - [6.3. Matching the DNS Domain Name Portion](#)
 - [6.4. Matching an IP Address Portion](#)
 - [6.5. Matching the Application Service Type Portion](#)
 - [6.6. Outcome](#)
- [7. Security Considerations](#)
 - [7.1. Wildcard Certificates](#)
 - [7.2. Uniform Resource Identifiers](#)
 - [7.3. Internationalized Domain Names](#)
 - [7.4. IP Addresses](#)
 - [7.5. Multiple Presented Identifiers](#)
 - [7.6. Multiple Reference Identifiers](#)
- [8. IANA Considerations](#)

[9. References](#)

[9.1. Normative References](#)

[9.2. Informative References](#)

[Appendix A. Changes from RFC 6125](#)

[Appendix B. Contributors](#)

[Acknowledgements](#)

[Authors' Addresses](#)

1. Introduction

1.1. Motivation

The visible face of the Internet largely consists of services that employ a client-server architecture in which a client communicates with an application service. When a client communicates with an application service using [\[TLS\]](#), [\[DTLS\]](#), or a protocol built on those ([\[QUIC\]](#) being a notable example), it has some notion of the server's identity (e.g., "the website at example.com") while attempting to establish secure communication. Likewise, during TLS negotiation, the server presents its notion of the service's identity in the form of a public-key certificate that was issued by a certificate authority (CA) in the context of the Internet Public Key Infrastructure using X.509 [\[PKIX\]](#). Informally, we can think of these identities as the client's "reference identity" and the server's "presented identity"; more formal definitions are given later. A client needs to verify that the server's presented identity matches its reference identity so it can deterministically and automatically authenticate the communication.

This document defines procedures for how clients do this verification. It therefore also defines requirements on other parties, such as the certificate authorities that issue certificates, the service administrators requesting them, and the protocol designers defining how things are named.

This document obsoletes RFC 6125. Changes from RFC 6125 are described under [Appendix A](#).

1.2. Applicability

This document does not supersede the rules for certificate issuance or validation specified by [\[PKIX\]](#). That document also governs any certificate-related topic on which this document is silent. This includes certificate syntax, extensions such as name constraints or extended key usage, and handling of certification paths.

This document addresses only name forms in the leaf "end entity" server certificate. It does not address the name forms in the chain of certificates used to validate a certificate, let alone creating or checking the validity of such a chain. In order to ensure proper

authentication, applications need to verify the entire certification path.

1.3. Overview of Recommendations

The previous version of this specification, [\[VERIFY\]](#), surveyed the then-current practice from many IETF standards and tried to generalize best practices (see Appendix A of [\[VERIFY\]](#) for details).

This document takes the lessons learned since then and codifies them. The following is a summary of the rules, which are described at greater length in the remainder of this document:

- *Only check DNS domain names via the subjectAlternativeName extension designed for that purpose: `dnsName`.

- *Allow use of even more specific subjectAlternativeName extensions where appropriate such as `uniformResourceIdentifier`, `iPAddress`, and the `otherName` form `SRVName`.

- *Wildcard support is now the default in certificates. Constrain wildcard certificates so that the wildcard can only be the complete left-most component of a domain name.

- *Do not include or check strings that look like domain names in the subject's Common Name.

1.4. Scope

1.4.1. In Scope

This document applies only to service identities that are used in TLS or DTLS and that are included in PKIX certificates.

With regard to TLS and DTLS, these security protocols are used to protect data exchanged over a wide variety of application protocols, which use both the TLS or DTLS handshake protocol and the TLS or DTLS record layer, either directly or through a profile as in Network Time Security [\[NTS\]](#). The TLS handshake protocol can also be used with different record layers to define secure transport protocols; at present the most prominent example is QUIC [\[RFC9000\]](#). The rules specified here are intended to apply to all protocols in this extended TLS "family".

With regard to PKIX certificates, the primary usage is in the context of the public key infrastructure described in [\[PKIX\]](#). In addition, technologies such as DNS-Based Authentication of Named Entities (DANE) [\[DANE\]](#) sometimes use certificates based on PKIX (more precisely, certificates structured via [\[X.509\]](#) or specific encodings thereof such as [\[X.690\]](#)), at least in certain modes. Alternatively, a

TLS peer could issue delegated credentials that are based on a CA-issued certificate, as in [[TLS-SUBCERTS](#)]. In both cases, a TLS client could learn of a service identity through its inclusion in the relevant certificate. The rules specified here are intended to apply whenever service identities are included in X.509 certificates or credentials that are derived from such certificates.

1.4.2. Out of Scope

The following topics are out of scope for this specification:

- *Security protocols other than those described above.
- *Keys or certificates employed outside the context of PKIX-based systems.
- *Client or end-user identities. Certificates representing client identities other than as described above, such as rfc822Name, are beyond the scope of this document.
- *Identification of servers using other than a domain name, IP address, or SRV service name. This document discusses Uniform Resource Identifiers [[URI](#)] only to the extent that they are expressed in certificates. Other aspects of a service such as a specific resource (the URI "path" component) or parameters (the URI "query" component) are the responsibility of specific protocols or URI schemes.
- *Certification authority policies. This includes items such as the following:
 - How to certify or validate fully-qualified domain names (FQDNs) and application service types (see [[ACME](#)] for some definition of this).
 - Types or "classes" of certificates to issue and whether to apply different policies for them.
 - How to certify or validate other kinds of information that might be included in a certificate (e.g., organization name).
- *Resolution of DNS domain names. Although the process whereby a client resolves the DNS domain name of an application service can involve several steps, for our purposes we care only about the fact that the client needs to verify the identity of the entity with which it communicates as a result of the resolution process. Thus, the resolution process itself is out of scope for this specification.

*User interface issues. In general, such issues are properly the responsibility of client software developers and standards development organizations dedicated to particular application technologies (see, for example, [[WSC-UI](#)]).

1.5. Terminology

Because many concepts related to "identity" are often too vague to be actionable in application protocols, we define a set of more concrete terms for use in this specification.

application service: A service on the Internet that enables clients to connect for the purpose of retrieving or uploading information, communicating with other entities, or connecting to a broader network of services.

application service provider: An entity that hosts or deploys an application service.

application service type: A formal identifier for the application protocol used to provide a particular kind of application service at a domain. This often appears as a URI scheme [[URI](#)], DNS SRV Service [[DNS-SRV](#)], or an ALPN [[ALPN](#)] identifier.

delegated domain: A domain name or host name that is explicitly configured at the application layer for communicating with the source domain (e.g., by the human user controlling an application client, by a trusted administrator who sets policy for such clients, or by means of automated configuration in the client itself). For example, an IMAP server at mail.example.net could be a delegated domain for a source domain of example.net associated with an email address of user@example.net. This kind of application-layer delegation is not to be confused with delegation in the DNS, by which a separate zone is created in the name space beneath the apex of a given domain; see for instance [[DNS-TERMS](#)].

derived domain: A domain name or host name that a client has derived from the source domain in an automated fashion (e.g., by means of an MX or SRV lookup).

identifier: A particular instance of an identifier type that is either presented by a server in a certificate or referenced by a client for matching purposes.

identifier type: A formally defined category of identifier that can be included in a certificate and therefore that can also be used

for matching purposes. For conciseness and convenience, we define the following identifier types of interest:

*DNS-ID: a subjectAltName entry of type `dnsName` as defined in [\[PKIX\]](#).

*IP-ID: a subjectAltName entry of type `iPAddress` as defined in [\[PKIX\]](#).

*SRV-ID: a subjectAltName entry of type `otherName` whose name form is `SRVName`, as defined in [\[SRVNAME\]](#).

*URI-ID: a subjectAltName entry of type `uniformResourceIdentifier` as defined in [\[PKIX\]](#). See further discussion in [Section 7.2](#).

PKIX: The short name for the Internet Public Key Infrastructure using X.509 defined in [\[PKIX\]](#). That document provides a profile of the X.509v3 certificate specifications and X.509v2 certificate revocation list (CRL) specifications for use on the Internet.

presented identifier: An identifier presented by a server to a client within a PKIX certificate when the client attempts to establish secure communication with the server. The certificate can include one or more presented identifiers of different types, and if the server hosts more than one domain then the certificate might present distinct identifiers for each domain.

reference identifier: An identifier used by the client when examining presented identifiers. It is constructed from the source domain, and optionally an application service type.

Relative Distinguished Name (RDN): An ASN.1-based construction which itself is a building-block component of Distinguished Names. See [\[LDAP-DN\]](#), [Section 2](#).

source domain: The fully qualified domain name (FQDN) that a client expects an application service to present in the certificate. This is typically input by a human user, configured into a client, or provided by reference such as a URL. The combination of a source domain and, optionally, an application service type enables a client to construct one or more reference identifiers. This

specification covers FQDNs only and provides no support for bare hostnames or any other name that does not include all labels.

subjectAltName entry: An identifier placed in a subjectAltName extension.

subjectAltName extension: A standard PKIX extension enabling identifiers of various types to be bound to the certificate subject.

subjectName: The name of a PKIX certificate's subject, encoded in a certificate's subject field (see [PKIX], [Section 4.1.2.6](#)).

TLS uses the words client and server, where the client is the entity that initiates the connection. In many cases, this is consistent with common practice, such as a browser connecting to a Web origin. For the sake of clarity, and to follow the usage in [TLS] and related specifications, we will continue to use the terms client and server in this document. However, these are TLS-layer roles, and the application protocol could support the TLS server making requests to the TLS client after the TLS handshake; there is no requirement that the roles at the application layer match the TLS layer.

Security-related terms used in this document, but not defined here or in [PKIX] should be understood in the sense defined in [SECTERMS]. Such terms include "attack", "authentication", "identity", "trust", "validate", and "verify".

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Identifying Application Services

This document assumes that an application service is identified by a DNS domain name (e.g., example.com), an IP address (IPv4 or IPv6), or by an identifier that contains additional supplementary information. Supplementary information is limited to the application service type as expressed in SRV (e.g., "the IMAP server at example.net") or a URI.

In a DNS-ID - and in the DNS domain name portion of an SRV-ID or URI-ID - any characters outside the [US-ASCII] range are prohibited and internationalized domain labels are represented as A-labels [IDNA-DEFS].

An IP address is either a 4-octet IPv4 address [[IPv4](#)] or a 16-octet IPv6 address [[IPv6](#)]. The identifier might need to be converted from a textual representation to obtain this value.

From the perspective of the application client or user, some identifiers are *direct* because they are provided directly by a human user. This includes runtime input, prior configuration, or explicit acceptance of a client communication attempt. Other names are *indirect* because they are automatically resolved by the application based on user input, such as a target name resolved from a source name using DNS SRV or [[NAPTR](#)] records. The distinction matters most for certificate consumption, specifically verification as discussed in this document.

From the perspective of the application service, some identifiers are *unrestricted* because they can be used in any type of service, such as a single certificate being used for both the HTTP and IMAP services at the host "example.com". Other identifiers are *restricted* because they can only be used for one type of service, such as a special-purpose certificate that can only be used for an IMAP service. This distinction matters most for certificate issuance.

We can categorize the four identifier types as follows:

- *A DNS-ID is direct and unrestricted.

- *An IP-ID is direct and unrestricted.

- *An SRV-ID is typically indirect but can be direct, and is restricted.

- *A URI-ID is direct and restricted.

It is important to keep these distinctions in mind, because best practices for the deployment and use of the identifiers differ. Note that cross-protocol attacks such as [[ALPACA](#)] are possible when two different protocol services use the same certificate. This can be addressed by using restricted identifiers or deploying services so that they do not share certificates. Protocol specifications **MUST** specify which identifiers are mandatory-to-implement and **SHOULD** provide operational guidance when necessary.

The Common Name RDN **MUST NOT** be used to identify a service because it is not strongly typed (essentially free-form text) and therefore suffers from ambiguities in interpretation.

For similar reasons, other RDNs within the subjectName **MUST NOT** be used to identify a service.

An IP address that is the result of a DNS query is not direct. Use of IP-IDs that are not direct is out of scope for this document.

3. Designing Application Protocols

This section defines how protocol designers should reference this document, which would typically be a normative reference in their specification. Its specification **MAY** choose to allow only one of the identifier types defined here.

If the technology does not use DNS SRV records to resolve the DNS domain names of application services, then its specification **MUST** state that SRV-ID as defined in this document is not supported. Note that many existing application technologies use DNS SRV records to resolve the DNS domain names of application services, but do not rely on representations of those records in PKIX certificates by means of SRV-IDs as defined in [[SRVNAME](#)].

If the technology does not use URIs to identify application services, then its specification **MUST** state that URI-ID as defined in this document is not supported. Note that many existing application technologies use URIs to identify application services, but do not rely on representation of those URIs in PKIX certificates by means of URI-IDs.

A technology **MAY** disallow the use of the wildcard character in presented identifiers. If it does so, then the specification **MUST** state that wildcard certificates as defined in this document are not supported.

A protocol can allow the use of an IP address in place of a DNS name. This might use the same field without distinguishing the type of identifier, as for example in the "host" components of a URI. In this case, applications need to be aware that the textual representation of an IPv4 address can appear to be a valid DNS name, even though it is not; the two types can be distinguished by first testing if the identifier is a valid IPv4 address, as is done by the "first-match-wins" algorithm in [Section 3.2.2](#) of [[URI](#)]. Note also that by policy, Top-Level Domains ([[DNS-TERMS](#)]) do not start with a digit (see [Section 2.2.1.3.2](#) of [[ICANN-AGB](#)]); historically this rule was also intended to apply to all labels in a domain name (see [Section 2.3.1](#) of [[DNS-NAMES](#)]), although that is not always the case in practice.

4. Representing Server Identity

This section provides instructions for issuers of certificates.

4.1. Rules

When a certificate authority issues a certificate based on the FQDN at which the application service provider will provide the relevant application, the following rules apply to the representation of application service identities. Note that some of these rules are cumulative and can interact in important ways that are illustrated later in this document.

1. The certificate **MUST** include at least one identifier.
2. The certificate **SHOULD** include a DNS-ID as a baseline for interoperability. This is not mandatory because it is legitimate for a certificate to include only an SRV-ID or URI-ID so as to scope its use to a particular application type.
3. If the service using the certificate deploys a technology for which the relevant specification stipulates that certificates should include identifiers of type "SRV-ID" (e.g., this is true of [\[XMPP\]](#)), then the certificate **SHOULD** include an SRV-ID. This identifier type could supplement the DNS-ID, unless the certificate is meant to be scoped to only the protocol in question.
4. If the service using the certificate deploys a technology for which the relevant specification stipulates that certificates should include identifiers of type URI-ID (e.g., this is true of [\[SIP\]](#) as specified by [\[SIP-CERTS\]](#)), then the certificate **SHOULD** include a URI-ID. The scheme **MUST** be that of the protocol associated with the application service type and the "host" component **MUST** be the FQDN of the service. The application protocol specification **MUST** specify which URI schemes are acceptable in URI-IDs contained in PKIX certificates used for the application protocol (e.g., sip but not sips or tel for SIP as described in [\[SIP-SIPS\]](#)). Typically this identifier type would supplement the DNS-ID, unless the certificate is meant to be scoped to only the protocol in question.
5. The certificate **MAY** contain more than one DNS-ID, SRV-ID, URI-ID, or IP-ID as further explained under [Section 7.5](#).
6. The certificate **MAY** include other application-specific identifiers for compatibility with a deployed base, especially identifiers for types that were defined before publication of [\[SRVNAME\]](#) or for which SRV service names or URI schemes do not exist. Such identifiers are out of scope for this specification.

4.2. Examples

Consider a simple website at `www.example.com`, which is not discoverable via DNS SRV lookups. Because HTTP does not specify the use of URIs in server certificates, a certificate for this service might include only a DNS-ID of `www.example.com`.

Consider the same website, which is reachable by a fixed IP address of `2001:db8::5c`. The certificate might include this value in an IP-ID to allow clients to use the fixed IP address as a reference identity.

Consider an IMAP-accessible email server at the host `mail.example.net` servicing email addresses of the form `user@example.net` and discoverable via DNS SRV lookups on the application service name of `example.net`. A certificate for this service might include SRV-IDs of `_imap.example.net` and `_imaps.example.net` (see [[EMAIL-SRV](#)]) along with DNS-IDs of `example.net` and `mail.example.net`.

Consider a SIP-accessible voice-over-IP (VoIP) server at the host `voice.example.edu` servicing SIP addresses of the form `user@voice.example.edu` and identified by a URI of `<sip:voice.example.edu>`. A certificate for this service would include a URI-ID of `sip:voice.example.edu` (see [[SIP-CERTS](#)]) along with a DNS-ID of `voice.example.edu`.

Consider an XMPP-compatible instant messaging (IM) server at the host `im.example.org` servicing IM addresses of the form `user@im.example.org` and discoverable via DNS SRV lookups on the `im.example.org` domain. A certificate for this service might include SRV-IDs of `_xmpp-client.im.example.org` and `_xmpp-server.im.example.org` (see [[XMPP](#)]), a DNS-ID of `im.example.org`.

5. Requesting Server Certificates

This section provides instructions for service providers regarding the information to include in certificate signing requests (CSRs). In general, service providers **SHOULD** request certificates that include all the identifier types that are required or recommended for the application service type that will be secured using the certificate to be issued.

A service provider **SHOULD** request certificates with as few identifiers as necessary to identify a single service; see [Section 7.5](#).

If the certificate will be used for only a single type of application service, the service provider **SHOULD** request a certificate that includes DNS-ID or IP-ID values that identify that service or, if appropriate for the application service type, SRV-ID or URI-ID values

that limit the deployment scope of the certificate to only the defined application service type.

If the certificate might be used for any type of application service, then the service provider **SHOULD** request a certificate that includes only DNS-IDs or IP-IDs. Again, because of multi-protocol attacks this practice is discouraged; this can be mitigated by deploying only one service on a host.

If a service provider offers multiple application service types and wishes to limit the applicability of certificates using SRV-IDs or URI-IDs, they **SHOULD** request multiple certificates, rather than a single certificate containing multiple SRV-IDs or URI-IDs each identifying a different application service type. This rule does not apply to application service type "bundles" that identify distinct access methods to the same underlying application such as an email application with access methods denoted by the application service types of imap, imaps, pop3, pop3s, and submission as described in [[EMAIL-SRV](#)].

6. Verifying Service Identity

At a high level, the client verifies the application service's identity by performing the following actions:

1. The client constructs a list of acceptable reference identifiers based on the source domain and, optionally, the type of service to which the client is connecting.
2. The server provides its identifiers in the form of a PKIX certificate.
3. The client checks each of its reference identifiers against the presented identifiers for the purpose of finding a match. When checking a reference identifier against a presented identifier, the client matches the source domain of the identifiers and, optionally, their application service type.

Naturally, in addition to checking identifiers, a client should perform further checks, such as expiration and revocation, to ensure that the server is authorized to provide the requested service. Because such checking is not a matter of verifying the application service identity presented in a certificate, methods for doing so are out of scope for this document.

6.1. Constructing a List of Reference Identifiers

6.1.1. Rules

The client **MUST** construct a list of acceptable reference identifiers, and **MUST** do so independently of the identifiers presented by the service.

The inputs used by the client to construct its list of reference identifiers might be a URI that a user has typed into an interface (e.g., an HTTPS URL for a website), configured account information (e.g., the domain name of a host for retrieving email, which might be different from the DNS domain name portion of a username), a hyperlink in a web page that triggers a browser to retrieve a media object or script, or some other combination of information that can yield a source domain and an application service type.

This document does not precisely define how reference identifiers are generated. Defining reference identifiers is the responsibility of applications or protocols that use this document. Because the security of a system that uses this document will depend on how reference identifiers are generated, great care should be taken in this process. For example, a protocol or application could specify that the application service type is obtained through a one-to-one mapping of URI schemes to service types or support only a restricted set of URI schemes. Similarly, it could insist that a domain name or IP address taken as input to the reference identifier must be obtained in a secure context such as a hyperlink embedded in a web page that was delivered over an authenticated and encrypted channel (see for instance [[SECURE-CONTEXTS](#)] with regard to the web platform).

Naturally, if the inputs themselves are invalid or corrupt (e.g., a user has clicked a hyperlink provided by a malicious entity in a phishing attack), then the client might end up communicating with an unexpected application service.

During the course of processing, a client might be exposed to identifiers that look like but are not reference identifiers. For example, DNS resolution that starts at a DNS-ID reference identifier might produce intermediate domain names that need to be further resolved. Any intermediate values are not reference identifiers and **MUST NOT** be treated as such, except as defined by the application. In the DNS case, not treating intermediate domain names as reference identifiers removes DNS and DNS resolution from the attack surface. However, an application might define a process for authenticating these intermediate identifiers in a way that then allows them to be used as a reference identifier; see for example [[SMTP-TLS](#)].

As one example of the process of generating a reference identifier, from user input of the URI <sip:alice@example.net> a client could derive the application service type sip from the URI scheme and parse the domain name example.net from the host component.

Using the combination of FQDN(s) or IP address(es), plus optionally an application service type, the client **MUST** construct its list of reference identifiers in accordance with the following rules:

*If a server for the application service type is typically associated with a URI for security purposes (i.e., a formal protocol document specifies the use of URIs in server certificates), then the reference identifier **SHOULD** be a URI-ID.

*If a server for the application service type is typically discovered by means of DNS SRV records, then the reference identifier **SHOULD** be an SRV-ID.

*If the reference identifier is an IP address, the reference identifier is an IP-ID.

*In the absence of more specific identifiers, the reference identifier is a DNS-ID. A reference identifier of type DNS-ID can be directly constructed from a FQDN that is (a) contained in or securely derived from the inputs, or (b) explicitly associated with the source domain by means of user configuration.

Which identifier types a client includes in its list of reference identifiers, and their priority, is a matter of local policy. For example, a client that is built to connect only to a particular kind of service might be configured to accept as valid only certificates that include an SRV-ID for that application service type. By contrast, a more lenient client, even if built to connect only to a particular kind of service, might include SRV-IDs, DNS-IDs, and IP-IDs in its list of reference identifiers.

6.1.2. Examples

The following examples are for illustrative purposes only and are not intended to be comprehensive.

1. A web browser that is connecting via HTTPS to the website at <https://www.example.com/> would have a single reference identifier: a DNS-ID of `www.example.com`.
2. A web browser connecting to <https://192.0.2.107/> would have a single IP-ID reference identifier of `192.0.2.107`.
3. A mail user agent that is connecting via IMAPS to the email service at `example.net` (resolved as `mail.example.net`) might have

three reference identifiers: an SRV-ID of `_imaps.example.net` (see [\[EMAIL-SRV\]](#)), and DNS-IDs of `example.net` and `mail.example.net`. An email user agent that does not support [\[EMAIL-SRV\]](#) would probably be explicitly configured to connect to `mail.example.net`, whereas an SRV-aware user agent would derive `example.net` from an email address of the form `user@example.net` but might also accept `mail.example.net` as the DNS domain name portion of reference identifiers for the service.

4. A voice-over-IP (VoIP) user agent that is connecting via SIP to the voice service at `voice.example.edu` might have only one reference identifier: a URI-ID of `sip:voice.example.edu` (see [\[SIP-CERTS\]](#)).
5. An instant messaging (IM) client that is connecting via XMPP to the IM service at `im.example.org` might have three reference identifiers: an SRV-ID of `_xmpp-client.im.example.org` (see [\[XMPP\]](#)), a DNS-ID of `im.example.org`, and an XMPP-specific `XmppAddr` of `im.example.org` (see [\[XMPP\]](#)).

In all these cases, presented identifiers that do not match the reference identifier(s) would be rejected; for instance:

*With regard to the first example a DNS-ID of `"web.example.com"` would be rejected because the DNS domain name portion does not match `"www.example.com"`.

*With regard to the third example, a URI-ID of `"sip:www.example.edu"` would be rejected because the DNS domain name portion does not match `"voice.example.edu"` and a DNS-ID of `"voice.example.edu"` would be rejected because it lacks the appropriate application service type portion (i.e., it does not specify a `"sip:"` URI).

6.2. Preparing to Seek a Match

Once the client has constructed its list of reference identifiers and has received the server's presented identifiers, the client checks its reference identifiers against the presented identifiers for the purpose of finding a match. The search fails if the client exhausts its list of reference identifiers without finding a match. The search succeeds if any presented identifier matches one of the reference identifiers, at which point the client **SHOULD** stop the search.

Before applying the comparison rules provided in the following sections, the client might need to split the reference identifier into components. Each reference identifier produces either a domain

name or an IP address and optionally an application service type as follows:

- *A DNS-ID reference identifier **MUST** be used directly as the DNS domain name and there is no application service type.
- *An IP-ID reference identifier **MUST** be exactly equal to the value of a `iPAddress` entry in `subjectAltName`, with no partial (e.g., network-level) matching. There is no application service type.
- *For an SRV-ID reference identifier, the DNS domain name portion is the `Name` and the application service type portion is the `Service`. For example, an SRV-ID of `_imaps.example.net` has a DNS domain name portion of `example.net` and an application service type portion of `imaps`, which maps to the IMAP application protocol as explained in [\[EMAIL-SRV\]](#).
- *For a reference identifier of type URI-ID, the DNS domain name portion is the "reg-name" part of the "host" component and the application service type portion is the scheme, as defined above. Matching only the "reg-name" rule from [\[URI\]](#) limits the additional domain name validation ([Section 6.3](#)) to DNS domain names or non-IP hostnames. A URI that contains an IP address might be matched against an IP-ID in place of a URI-ID by some lenient clients. This document does not describe how a URI that contains no "host" component can be matched. Note that extraction of the "reg-name" might necessitate normalization of the URI (as explained in [Section 6](#) of [\[URI\]](#)). For example, a URI-ID of `sip:voice.example.edu` would be split into a DNS domain name portion of `voice.example.edu` and an application service type of `sip` (associated with an application protocol of SIP as explained in [\[SIP-CERTS\]](#)).

If the reference identifier produces a domain name, the client **MUST** match the DNS name; see [Section 6.3](#). If the reference identifier produces an IP address, the client **MUST** match the IP address; see [Section 6.4](#). If an application service type is present it **MUST** also match the service type as well; see [Section 6.5](#).

6.3. Matching the DNS Domain Name Portion

This section describes how the client must determine if the presented DNS name matches the reference DNS name. The rules differ depending on whether the domain to be checked is a traditional domain name or an internationalized domain name, as defined in [Section 2](#). For clients that support presented identifiers containing the wildcard character "*", this section also specifies a supplemental rule for such "wildcard certificates". This section uses the description of labels and domain names in [\[DNS-CONCEPTS\]](#).

If the DNS domain name portion of a reference identifier is a "traditional domain name" (i.e., a FQDN that conforms to "preferred name syntax" as described in [Section 3.5](#) of [\[DNS-CONCEPTS\]](#)), then matching of the reference identifier against the presented identifier **MUST** be performed by comparing the set of domain name labels using a case-insensitive ASCII comparison, as clarified by [\[DNS-CASE\]](#). For example, `WWW.Example.Com` would be lower-cased to `www.example.com` for comparison purposes. Each label **MUST** match in order for the names to be considered to match, except as supplemented by the rule about checking of wildcard labels in presented identifiers given below.

If the DNS domain name portion of a reference identifier is an internationalized domain name, then the client **MUST** convert any U-labels [\[IDNA-DEFS\]](#) in the domain name to A-labels before checking the domain name or comparing it with others. In accordance with [\[IDNA-PROTO\]](#), A-labels **MUST** be compared as case-insensitive ASCII. Each label **MUST** match in order for the domain names to be considered to match, except as supplemented by the rule about checking of wildcard labels in presented identifiers given below.

If the technology specification supports wildcards in presented identifiers, then the client **MUST** match the reference identifier against a presented identifier whose DNS domain name portion contains the wildcard character "*" in a label provided these requirements are met:

1. There is only one wildcard character.
2. The wildcard character appears only as the complete content of the left-most label.

If the requirements are not met, the presented identifier is invalid and **MUST** be ignored.

A wildcard in a presented identifier can only match exactly one label in a reference identifier. This specification covers only wildcard characters in presented identifiers, not wildcard characters in reference identifiers or in DNS domain names more generally. Therefore the use of wildcard characters as described herein is not to be confused with DNS wildcard matching, where the "*" label always matches at least one whole label and sometimes more; see [\[DNS-CONCEPTS\]](#), [Section 4.3.3](#) and [\[DNS-WILDCARDS\]](#).

For information regarding the security characteristics of wildcard certificates, see [Section 7.1](#).

6.4. Matching an IP Address Portion

An IP-ID matches based on an octet-for-octet comparison of the bytes of the reference identity with the bytes contained in the `iPAddress`

subjectAltName. Because the ipAddress field does not include the IP version, a helpful heuristic for implementors is to distinguish IPv4 addresses from IPv6 addresses by their length.

For an IP address that appears in a URI-ID, the "host" component of both the reference identity and the presented identifier must match. These are parsed as either an "IP-literal" (following [[IPv6](#)]) or an "IPv4address" (following [[IPv4](#)]). If the resulting octets are equal, the IP address matches.

This document does not specify how an SRV-ID reference identity can include an IP address.

6.5. Matching the Application Service Type Portion

The rules for matching the application service type depend on whether the identifier is an SRV-ID or a URI-ID.

These identifiers provide an application service type portion to be checked, but that portion is combined only with the DNS domain name portion of the SRV-ID or URI-ID itself. For example, if a client's list of reference identifiers includes an SRV-ID of _xmpp-client.im.example.org and a DNS-ID of apps.example.net, the client **MUST** check both the combination of an application service type of xmpp-client and a DNS domain name of im.example.org and, separately, a DNS domain name of apps.example.net. However, the client **MUST NOT** check the combination of an application service type of xmpp-client and a DNS domain name of apps.example.net because it does not have an SRV-ID of _xmpp-client.apps.example.net in its list of reference identifiers.

If the identifier is an SRV-ID, then the application service name **MUST** be matched in a case-insensitive manner, in accordance with [[DNS-SRV](#)]. Note that, per [[SRVNAME](#)], the _ character is part of the service name in DNS SRV records and in SRV-IDs.

If the identifier is a URI-ID, then the scheme name portion **MUST** be matched in a case-insensitive manner, in accordance with [[URI](#)]. Note that the : character is a separator between the scheme name and the rest of the URI, and thus does not need to be included in any comparison.

6.6. Outcome

If the client has found a presented identifier that matches a reference identifier, then the service identity check has succeeded. In this case, the client **MUST** use the matched reference identifier as the validated identity of the application service.

If the client does not find a presented identifier matching any of the reference identifiers, then the client **MUST** proceed as described as follows.

If the client is an automated application, then it **SHOULD** terminate the communication attempt with a bad certificate error and log the error appropriately. The application **MAY** provide a configuration setting to disable this behavior, but it **MUST** enable it by default.

If the client is one that is directly controlled by a human user, then it **SHOULD** inform the user of the identity mismatch and automatically terminate the communication attempt with a bad certificate error in order to prevent users from inadvertently bypassing security protections in hostile situations. Such clients **MAY** give advanced users the option of proceeding with acceptance despite the identity mismatch. Although this behavior can be appropriate in certain specialized circumstances, it needs to be handled with extreme caution, for example by first encouraging even an advanced user to terminate the communication attempt and, if they choose to proceed anyway, by forcing the user to view the entire certification path before proceeding.

The application **MAY** also present the user with the ability to accept the presented certificate as valid for subsequent connections. Such ad-hoc "pinning" **SHOULD NOT** restrict future connections to just the pinned certificate. Local policy that statically enforces a given certificate for a given peer **SHOULD** be made available only as prior configuration, rather than a just-in-time override for a failed connection.

7. Security Considerations

7.1. Wildcard Certificates

Wildcard certificates automatically vouch for any single-label host names within their domain, but not multiple levels of domains. This can be convenient for administrators but also poses the risk of vouching for rogue or buggy hosts. See for example [[Defeating-SSL](#)] (beginning at slide 91) and [[HTTPSbytes](#)] (slides 38-40).

As specified in [Section 6.3](#), restricting the presented identifiers in certificates to only one wildcard character (e.g., `*.example.com` but not `*.*.example.com`) and restricting the use of wildcards to only the left-most domain label can help to mitigate certain aspects of the attack described in [[Defeating-SSL](#)].

That same attack also relies on the initial use of a cleartext HTTP connection, which is hijacked by an active on-path attacker and subsequently upgraded to HTTPS. In order to mitigate such an attack,

administrators and software developers are advised to follow the strict TLS guidelines provided in [\[TLS-REQS\]](#), [Section 3.2](#).

Because the attack described in [\[HTTPSbytes\]](#) relies on an underlying cross-site scripting (XSS) attack, web browsers and applications are advised to follow best practices to prevent XSS attacks; see for example [\[XSS\]](#) published by the Open Web Application Security Project (OWASP).

Protection against a wildcard that identifies a public suffix [\[Public-Suffix\]](#), such as *.co.uk or *.com, is beyond the scope of this document.

As noted in [Section 3](#), application protocols can disallow the use of wildcard certificates entirely as a more foolproof mitigation.

7.2. Uniform Resource Identifiers

The URI-ID type is a subjectAltName entry of type uniformResourceIdentifier as defined in [\[PKIX\]](#). For the purposes of this specification, the URI-ID **MUST** include both a "scheme" and a "host" component that matches the "reg-name" rule; if the entry does not include both, it is not a valid URI-ID and **MUST** be ignored. Any other components are ignored, because only the "scheme" and "host" components are used for certificate matching as specified under [Section 6](#).

The quoted component names in the previous paragraph represent the associated [\[ABNF\]](#) productions from the IETF standard for Uniform Resource Identifiers [\[URI\]](#). Although the reader should be aware that some applications (e.g., web browsers) might instead conform to the Uniform Resource Locator (URL) specification maintained by the WHATWG [\[URL\]](#), it is not expected that differences between the URI and URL specifications would manifest themselves in certificate matching.

7.3. Internationalized Domain Names

This document specifies only matching between reference identifiers and presented identifiers, not the visual presentation of domain names. More specifically, matching of internationalized domain names is performed on A-labels only [Section 6](#). The limited scope of this specification likely mitigates potential confusion caused by the use of visually similar characters in domain names (as described for example in [\[IDNA-DEFS\]](#), [Section 4.4](#), [\[UTS-36\]](#), and [\[UTS-39\]](#)); in any case, such concerns are a matter for application-level protocols and user interfaces, not the matching of certificates.

7.4. IP Addresses

The TLS Server Name Indication (SNI) extension only conveys domain names. Therefore, a client with an IP-ID reference identity cannot present any information about its reference identity when connecting to a server. Servers that wish to present an IP-ID therefore need to present this identity when a connection is made without SNI.

The textual representation of an IPv4 address might be misinterpreted as a valid FQDN in some contexts. This can result in different security treatment that might cause different components of a system to classify the value differently, which might lead to vulnerabilities. For example, one system component enforces a security rule that is conditional on the type of identifier. This component misclassifies an IP address as an FQDN. A different component correctly classifies the identifier but might incorrectly assume that rules regarding IP addresses have been enforced. Consistent classification of identifiers avoids this problem.

7.5. Multiple Presented Identifiers

A given application service might be addressed by multiple DNS domain names for a variety of reasons, and a given deployment might service multiple domains or protocols. TLS Extensions such as TLS Server Name Indication (SNI), discussed in [TLS], [Section 4.4.2.2](#), and Application Layer Protocol Negotiation (ALPN), discussed in [ALPN], provide a way for the application to indicate the desired identifier and protocol to the server, which it can then use to select the most appropriate certificate.

This specification allows multiple DNS-IDs, IP-IDs, SRV-IDs, or URI-IDs in a certificate. As a result, an application service can use the same certificate for multiple hostnames, such as when a client does not support the TLS SNI extension, or for multiple protocols, such as SMTP and HTTP, on a single hostname. Note that the set of names in a certificate is the set of names that could be affected by a compromise of any other server named in the set: the strength of any server in the set of names is determined by the weakest of those servers that offer the names.

Methods for mitigating this risk includes: limiting the number of names that any server can speak for, following the guidelines for use of [ALPN] described in Section 3.8 of [TLS-REQS]), and ensuring that all servers in the set have a strong minimum configuration as described in Section 3.9 of [TLS-REQS].

7.6. Multiple Reference Identifiers

This specification describes how a client may construct multiple acceptable reference identifiers and may match any of those reference

identifiers with the set of presented identifiers. [PKIX], [Section 4.2.1.10](#) describes a mechanism to allow CA certificates to be constrained in the set of presented identifiers that they may include within server certificates. However, these constraints only apply to the explicitly enumerated name forms. For example, a CA that is only name constrained for DNS-IDs is not constrained for SRV-IDs and URI-IDs, unless those name forms are also explicitly included within the name constraints extension.

A client that constructs multiple reference identifiers of different types, such as both DNS-ID and SRV-IDs, as described in [Section 6.1.1](#), **SHOULD** take care to ensure that CAs issuing such certificates are appropriately constrained. This **MAY** take the form of local policy through agreement with the issuing CA, or **MAY** be enforced by the client requiring that if one form of presented identifier is constrained, such as a `dnsName` name constraint for DNS-IDs, then all other forms of acceptable reference identities are also constrained, such as requiring a `uniformResourceIndicator` name constraint for URI-IDs.

8. IANA Considerations

This document has no actions for IANA.

9. References

9.1. Normative References

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

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

[DNS-SRV] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, DOI 10.17487/RFC2782, February 2000, <<https://www.rfc-editor.org/rfc/rfc2782>>.

[DNS-WILDCARDS] Lewis, E., "The Role of Wildcards in the Domain Name System", RFC 4592, DOI 10.17487/RFC4592, July 2006, <<https://www.rfc-editor.org/rfc/rfc4592>>.

[IDNA-DEFS] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <<https://www.rfc-editor.org/rfc/rfc5890>>.

[IDNA-PROTO]

Klensin, J., "Internationalized Domain Names in Applications (IDNA): Protocol", RFC 5891, DOI 10.17487/RFC5891, August 2010, <<https://www.rfc-editor.org/rfc/rfc5891>>.

[IPv4]

Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <<https://www.rfc-editor.org/rfc/rfc791>>.

[IPv6]

Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <<https://www.rfc-editor.org/rfc/rfc4291>>.

[LDAP-DN]

Zeilenga, K., Ed., "Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names", RFC 4514, DOI 10.17487/RFC4514, June 2006, <<https://www.rfc-editor.org/rfc/rfc4514>>.

[PKIX]

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, <<https://www.rfc-editor.org/rfc/rfc5280>>.

[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/rfc/rfc2119>>.

[RFC8174]

Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/rfc/rfc8174>>.

[SRVNAME]

Santesson, S., "Internet X.509 Public Key Infrastructure Subject Alternative Name for Expression of Service Name", RFC 4985, DOI 10.17487/RFC4985, August 2007, <<https://www.rfc-editor.org/rfc/rfc4985>>.

[TLS-REQS]

Sheffer, Y., Saint-Andre, P., and T. Fossati, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 9325, DOI 10.17487/RFC9325, November 2022, <<https://www.rfc-editor.org/rfc/rfc9325>>.

[URI]

Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <<https://www.rfc-editor.org/rfc/rfc3986>>.

9.2. Informative References

- [ABNF] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <<https://www.rfc-editor.org/rfc/rfc5234>>.
- [ACME] Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", RFC 8555, DOI 10.17487/RFC8555, March 2019, <<https://www.rfc-editor.org/rfc/rfc8555>>.
- [ALPACA] Brinkmann, M., Dresen, C., Merget, R., Poddebniak, D., Müller, J., Somorovsky, J., Schwenk, J., and S. Schinzel, "ALPACA: Application Layer Protocol Confusion - Analyzing and Mitigating Cracks in TLS Authentication", September 2021, <<https://alpaca-attack.com/ALPACA.pdf>>.
- [ALPN] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <<https://www.rfc-editor.org/rfc/rfc7301>>.
- [DANE] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <<https://www.rfc-editor.org/rfc/rfc6698>>.
- [Defeating-SSL] Marlinspike, M., "New Tricks for Defeating SSL in Practice", BlackHat DC, February 2009, <<http://www.blackhat.com/presentations/bh-dc-09/Marlinspike/BlackHat-DC-09-Marlinspike-Defeating-SSL.pdf>>.
- [DNS-CASE] Eastlake 3rd, D., "Domain Name System (DNS) Case Insensitivity Clarification", RFC 4343, DOI 10.17487/RFC4343, January 2006, <<https://www.rfc-editor.org/rfc/rfc4343>>.
- [DNS-TERMS] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", BCP 219, RFC 8499, DOI 10.17487/RFC8499, January 2019, <<https://www.rfc-editor.org/rfc/rfc8499>>.
- [DTLS] Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", RFC 9147, DOI 10.17487/RFC9147, April 2022, <<https://www.rfc-editor.org/rfc/rfc9147>>.
- [EMAIL-SRV] Daboo, C., "Use of SRV Records for Locating Email Submission/Access Services", RFC 6186, DOI 10.17487/

RFC6186, March 2011, <<https://www.rfc-editor.org/rfc/rfc6186>>.

[HTTP] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <<https://www.rfc-editor.org/rfc/rfc9110>>.

[HTTPSbytes] Sokol, J. and R. Hansen, "HTTPS Can Byte Me", BlackHat Abu Dhabi, November 2010, <<https://media.blackhat.com/bh-ad-10/Hansen/Blackhat-AD-2010-Hansen-Sokol-HTTPS-Can-Byte-Me-slides.pdf>>.

[ICANN-AGB] ICANN, "gTLD Applicant Guidebook", 4 June 2012, <<https://newgtlds.icann.org/en/applicants/agb>>.

[NAPTR] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database", RFC 3403, DOI 10.17487/RFC3403, October 2002, <<https://www.rfc-editor.org/rfc/rfc3403>>.

[NTS] Franke, D., Sibold, D., Teichel, K., Dansarie, M., and R. Sundblad, "Network Time Security for the Network Time Protocol", RFC 8915, DOI 10.17487/RFC8915, September 2020, <<https://www.rfc-editor.org/rfc/rfc8915>>.

[Public-Suffix] "Public Suffix List", 2020, <<https://publicsuffix.org>>.

[QUIC] Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <<https://www.rfc-editor.org/rfc/rfc9001>>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <<https://www.rfc-editor.org/rfc/rfc9000>>.

[SECTERMS] Shirey, R., "Internet Security Glossary, Version 2", FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007, <<https://www.rfc-editor.org/rfc/rfc4949>>.

[SECURE-CONTEXTS] West, M., "Secure Contexts", 2021, <<https://www.w3.org/TR/secure-contexts/>>.

[SIP] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261,

DOI 10.17487/RFC3261, June 2002, <<https://www.rfc-editor.org/rfc/rfc3261>>.

[SIP-CERTS] Gurbani, V., Lawrence, S., and A. Jeffrey, "Domain Certificates in the Session Initiation Protocol (SIP)", RFC 5922, DOI 10.17487/RFC5922, June 2010, <<https://www.rfc-editor.org/rfc/rfc5922>>.

[SIP-SIPS] Audet, F., "The Use of the SIPS URI Scheme in the Session Initiation Protocol (SIP)", RFC 5630, DOI 10.17487/RFC5630, October 2009, <<https://www.rfc-editor.org/rfc/rfc5630>>.

[SMTP-TLS] Fenton, J., "SMTP Require TLS Option", RFC 8689, DOI 10.17487/RFC8689, November 2019, <<https://www.rfc-editor.org/rfc/rfc8689>>.

[TLS] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/rfc/rfc8446>>.

[TLS-SUBCERTS] Barnes, R., Iyengar, S., Sullivan, N., and E. Rescorla, "Delegated Credentials for (D)TLS", Work in Progress, Internet-Draft, draft-ietf-tls-subcerts-15, 30 June 2022, <<https://datatracker.ietf.org/doc/html/draft-ietf-tls-subcerts-15>>.

[URL] van Kesteren, A., "URL", 2023, <<https://url.spec.whatwg.org/>>.

[US-ASCII] American National Standards Institute, "Coded Character Set - 7-bit American Standard Code for Information Interchange", ANSI X3.4, 1986.

[UTS-36] Davis, M. and M. Suignard, "Unicode Security Considerations", 2014, <<https://unicode.org/reports/tr36/>>.

[UTS-39] Davis, M. and M. Suignard, "Unicode Security Mechanisms", 2022, <<https://unicode.org/reports/tr39/>>.

[VERIFY] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer

Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <<https://www.rfc-editor.org/rfc/rfc6125>>.

- [WSC-UI] Saldhana, A. and T. Roessler, "Web Security Context: User Interface Guidelines", August 2010, <<https://www.w3.org/TR/2010/REC-wsc-ui-20100812/>>.
- [X.509] International Telecommunications Union, "Information Technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks", ITU-T X.509, 2005.
- [X.690] International Telecommunications Union, "Information Technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T X.690, 2008.
- [XMPP] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, DOI 10.17487/RFC6120, March 2011, <<https://www.rfc-editor.org/rfc/rfc6120>>.
- [XSS] OWASP, "Cross Site Scripting (XSS)", 2022, <<https://owasp.org/www-community/attacks/xss/>>.

Appendix A. Changes from RFC 6125

This document revises and obsoletes [VERIFY] based on the decade of experience and changes since it was published. The major changes, in no particular order, include:

- *The only legal place for a certificate wildcard is as the complete left-most component in a domain name.
- *The server identity can only be expressed in the subjectAltNames extension; it is no longer valid to use the commonName RDN, known as CN-ID in [VERIFY].
- *Detailed discussion of pinning (configuring use of a certificate that doesn't match the criteria in this document) has been removed and replaced with two paragraphs in [Section 6.6](#).
- *The sections detailing different target audiences and which sections to read (first) have been removed.
- *References to the X.500 directory, the survey of prior art, and the sample text in Appendix A have been removed.
- *All references have been updated to the current latest version.
- *The TLS SNI extension is no longer new, it is commonplace.

*Additional text on multiple identifiers, and their security considerations, has been added.

*IP-ID reference identifiers are added. This builds on the definition in [Section 4.3.5](#) of [\[HTTP\]](#).

Appendix B. Contributors

Jeff Hodges co-authored the previous version of these recommendations, [\[VERIFY\]](#). The authors gratefully acknowledge his essential contributions to this work.

Martin Thomson contributed the text on handling of IP-IDs.

Acknowledgements

We gratefully acknowledge everyone who contributed to the previous version of these recommendations, [\[VERIFY\]](#). Thanks also to Carsten Bormann for converting the previous document to Markdown so that we could more easily use Martin Thomson's i-d-template software.

In addition to discussion on the mailing list, the following people provided especially helpful feedback: Viktor Dukhovni, Jim Fenton, Olle Johansson, John Klensin, John Mattson, Alexey Melnikov, Yaron Sheffer, Ryan Sleevi, Brian Smith, and Martin Thomson.

A few descriptive sentences were borrowed from [\[TLS-REQS\]](#).

Authors' Addresses

Peter Saint-Andre
independent
United States of America

Email: stpeter@stpeter.im

Rich Salz
Akamai Technologies
United States of America

Email: rsalz@akamai.com