

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: May 3, 2020

G. Lozano
ICANN
Oct 31, 2019

**ICANN TMCH functional specifications
draft-ietf-regext-tmch-func-spec-05**

Abstract

This document describes the requirements, the architecture and the interfaces between the ICANN Trademark Clearinghouse (TMCH) and Domain Name Registries as well as between the ICANN TMCH and Domain Name Registrars for the provisioning and management of domain names during Sunrise and Trademark Claims Periods.

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 May 3, 2020.

Copyright Notice

Copyright (c) 2019 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 Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	3
2.	Terminology	4
3.	Glossary	4
4.	Architecture	9
4.1.	Sunrise Period	9
4.2.	Trademark Claims Period	10
4.3.	Interfaces	10
4.3.1.	hv	10
4.3.2.	vd	11
4.3.3.	dy	11
4.3.4.	tr	11
4.3.5.	ry	11
4.3.6.	dr	11
4.3.7.	yd	11
4.3.8.	dv	12
4.3.9.	vh	12
4.3.10.	vs	12
4.3.11.	sy	12
4.3.12.	sr	12
4.3.13.	vc	13
4.3.14.	cy	13
4.3.15.	cr	13
5.	Process Descriptions	13
5.1.	Bootstrapping	13
5.1.1.	Bootstrapping for Registries	13
5.1.1.1.	Credentials	13
5.1.1.2.	IP Addresses for Access Control	14
5.1.1.3.	ICANN TMCH Trust Anchor	14
5.1.1.4.	TMDB PGP Key	14
5.1.2.	Bootstrapping for Registrars	15
5.1.2.1.	Credentials	15
5.1.2.2.	IP Addresses for Access Control	15
5.1.2.3.	ICANN TMCH Trust Anchor	15
5.1.2.4.	TMDB PGP Key	15
5.2.	Sunrise Period	16
5.2.1.	Domain Name registration	16
5.2.2.	Sunrise Domain Name registration by Registries	17
5.2.3.	TMDB Sunrise Services for Registries	18
5.2.3.1.	SMD Revocation List	18
5.2.3.2.	TMV Certificate Revocation List (CRL)	18
5.2.3.3.	Notice of Registered Domain Names (NORN)	19
5.2.4.	Sunrise Domain Name registration by Registrars	22
5.2.5.	TMDB Sunrise Services for Registrars	22
5.3.	Trademark Claims Period	23
5.3.1.	Domain Registration	23
5.3.2.	Trademark Claims Domain Name registration by	

Registries	24
5.3.3. TMBD Trademark Claims Services for Registries	25
5.3.3.1. Domain Name Label (DNL) List	25
5.3.3.2. Notice of Registered Domain Names (NORN)	26
5.3.4. Trademark Claims Domain Name registration by Registrars	26
5.3.5. TMBD Trademark Claims Services for Registrars	28
5.3.5.1. Claims Notice Information Service (CNIS)	28
5.4. Qualified Launch Program (QLP) Period	28
5.4.1. Domain Registration	28
5.4.2. TMBD QLP Services for Registries	31
5.4.2.1. Sunrise List (SURL)	31
6. Data Format Descriptions	31
6.1. Domain Name Label (DNL) List	31
6.2. SMD Revocation List	33
6.3. List of Registered Domain Names (LORDN) file	35
6.3.1. LORDN Log file	40
6.3.1.1. LORDN Log Result Codes	42
6.4. Signed Mark Data (SMD) File	46
6.5. Trademark Claims Notice (TCN)	47
6.6. Sunrise List (SURL)	54
7. Formal Syntax	55
7.1. Trademark Claims Notice (TCN)	55
8. Acknowledgements	58
9. Change History	58
9.1. Version 04	58
9.2. Version 05	58
10. IANA Considerations	58
11. Security Considerations	59
12. References	59
12.1. Normative References	59
12.2. Informative References	60
Author's Address	62

[1.](#) Introduction

Domain Name Registries (DNRs) may operate in special modes for certain periods of time enabling trademark holders to protect their rights during the introduction of a Top Level Domain (TLD).

Along with the introduction of new generic TLDs (gTLD), two special modes came into effect:

- o Sunrise Period, the Sunrise Period allows trademark holders an advance opportunity to register domain names corresponding to their marks before names are generally available to the public.

- o Trademark Claims Period, the Trademark Claims Period follows the Sunrise Period and runs for at least the first 90 days of an initial operating period of general registration. During the Trademark Claims Period, anyone attempting to register a domain name matching a mark that is recorded in the ICANN Trademark Clearinghouse (TMCH) will receive a notification displaying the relevant mark information.

This document describes the requirements, the architecture and the interfaces between the ICANN TMCH and Domain Name Registries (called Registries in the rest of the document) as well as between the ICANN TMCH and Domain Name Registrars (called Registrars in the rest of the document) for the provisioning and management of domain names during the Sunrise and Trademark Claims Periods.

For any date and/or time indications, Coordinated Universal Time (UTC) applies.

2. Terminology

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](#)].

XML is case sensitive. Unless stated otherwise, XML specifications and examples provided in this document MUST be interpreted in the character case presented in order to develop a conforming implementation.

"tmNotice-1.0" is used as an abbreviation for "urn:ietf:params:xml:ns:tmNotice-1.0". The XML namespace prefix "tmNotice" is used, but implementations MUST NOT depend on it and instead employ a proper namespace-aware XML parser and serializer to interpret and output the XML documents.

3. Glossary

In the following section, the most common terms are briefly explained:

- o Backend Registry Operator: Entity that manages (a part of) the technical infrastructure for a Registry Operator. The Registry Operator may also be the Backend Registry Operator.
- o CA: Certificate Authority, see [[RFC5280](#)].
- o CNIS, Claims Notice Information Service: This service provides Trademark Claims Notices (TCN) to Registrars.

- o CRC32, Cyclic Redundancy Check: algorithm used in the ISO 3309 standard and in [section 8.1.1.6.2](#) of ITU-T recommendation V.42.
- o CRL: Certificate Revocation List, see [[RFC5280](#)].
- o CSV: Comma-Separated Values, see [[RFC4180](#)]
- o Date and time, datetime: Date and time are specified following the standard "Date and Time on the Internet specification", see [[RFC3339](#)].
- o DN, Domain Name, domain name: see definition of Domain name in [[RFC7719](#)].
- o DNROID, DN Repository Object IDentifier: an identifier assigned by the Registry to each DN object that unequivocally identifies said DN object. For example, if a new DN object is created for a name that existed in the past, the DN objects will have different DNRIDs.
- o DNL, Domain Name Label, the DNL is an A-label or NR-LDH label (see [[RFC5890](#)]).
- o DNL List: A list of DNLs that are covered by a PRM.
- o DNS: Domain Name System, see [[RFC7719](#)].
- o Effective allocation: A DN is considered effectively allocated when the DN object for the DN has been created in the SRS of the Registry and has been assigned to the effective user. A DN object in status "pendingCreate" or any other status that precedes the first time a DN is assigned to an end-user is not considered an effective allocation. A DN object created internally by the Registry for subsequent delegation to another Registrant is not considered an effective allocation.
- o EPP: The Extensible Provisioning Protocol, see definition of EPP in [[RFC7719](#)].
- o FQDN: Fully-Qualified Domain Name, see definition of FQDN in [[RFC7719](#)].
- o HTTP: Hypertext Transfer Protocol, see [[RFC7230](#)] and [[RFC7231](#)].
- o HTTPS: HTTP over TLS (Transport Layer Security), [[RFC2818](#)].
- o IDN: Internationalized Domain Name, see definition of IDN in [[RFC7719](#)].

- o Lookup Key: A random string of up to 51 chars from the set [a-zA-Z0-9/] to be used as the lookup key by Registrars to obtain the TCN using the CNIS. Lookup Keys are unique and are related to one DNL only.
- o LORDN, List of Registered Domain Names: This is the list of effectively allocated DNS matching a DNL of a PRM. Registries will upload this list to the TMDB (during the NORDN process).
- o Matching Rules: Some trademarks entitled to inclusion in the TMDB include characters that are impermissible in the domain name system (DNS) as a DNL. The TMV changes (using the ICANN TMCH Matching Rules [[MatchingRules](#)]) certain DNS-impermissible characters in a trademark into DNS-permissible equivalent characters
- o NORDN, Notification of Registered Domain Names: The process by which Registries upload their recent LORDN to the TMDB.
- o PGP: Pretty Good Privacy, see [[RFC4880](#)]
- o PKI: Public Key Infrastructure, see [[RFC5280](#)].
- o PRM, Pre-registered mark: Mark that has been pre-registered with the ICANN TMCH.
- o QLP Period, Qualified Launch Program Period: During this optional period, a special process applies to DNS matching the Sunrise List (SURL) and/or the DNL List, to ensure that TMHs are informed of a DN matching their PRM.
- o Registrant: see definition of Registrant in [[RFC7719](#)].
- o Registrar, Domain Name Registrar: see definition of Registrar in [[RFC7719](#)].
- o Registry, Domain Name Registry, Registry Operator: see definition of Registry in [[RFC7719](#)]. A Registry Operator is the contracting party with ICANN for the TLD.
- o SMD, Signed Mark Data: A cryptographically signed token issued by the TMV to the TMH to be used in the Sunrise Period to apply for a DN that matches a DNL of a PRM; see also [[RFC7848](#)]. An SMD generated by an ICANN-approved trademark validator (TMV) contains both the signed token and the TMV's PKIX certificate.

- o SMD File: A file containing the SMD (see above) and some human readable data. The latter is usually ignored in the processing of the SMD File. See also [Section 6.4](#).
- o SMD Revocation List: The SMD Revocation List is used by Registries (and optionally by Registrars) during the Sunrise Period to ensure that an SMD is still valid (i.e. not revoked). The SMD Revocation List has a similar function as CRLs used in PKI.
- o SRS: Shared Registration System, see also [\[ICANN-GTLD-AGB-20120604\]](#).
- o SURL, Sunrise List: The list of DNLs that are covered by a PRM and eligible for Sunrise.
- o Sunrise Period: During this period DNs matching a DNL of a PRM can be exclusively obtained by the respective TMHs. For DNs matching a PRM, a special process applies to ensure that TMHs are informed on the effective allocation of a DN matching their PRM.
- o TLD: Top-Level Domain Name, see definition of TLD in [\[RFC7719\]](#).
- o ICANN TMCH: a central repository for information to be authenticated, stored, and disseminated, pertaining to the rights of TMHs. The ICANN TMCH is split into two functions TMV and TMDB (see below). There could be several entities performing the TMV function, but only one entity performing the TMDB function.
- o ICANN TMCH-CA: The Certificate Authority (CA) for the ICANN TMCH. This CA is operated by ICANN. The public key for this CA is the trust anchor used to validate the identity of each TMV.
- o TMDB, Trademark Clearinghouse Database: Serves as a database of the ICANN TMCH to provide information to the gTLD Registries and Registrars to support Sunrise or Trademark Claims services. There is only one TMDB in the ICANN TMCH that concentrates the information about the "verified" Trademark records from the TMVs.
- o TMH, Trademark Holder: The person or organization owning rights on a mark.
- o TMV, Trademark Validator, Trademark validation organization: An entity authorized by ICANN to authenticate and validate registrations in the TMDB ensuring the marks qualify as registered or are court-validated marks or marks that are protected by statute or treaty. This entity would also be asked to ensure that proof of use of marks is provided, which can be demonstrated by furnishing a signed declaration and one specimen of current use.

- o Trademark, mark: Marks are used to claim exclusive properties of products or services. A mark is typically a name, word, phrase, logo, symbol, design, image, or a combination of these elements. For the scope of this document only textual marks are relevant.
- o Trademark Claims, Claims: Provides information to enhance the understanding of the Trademark rights being claimed by the TMH.
- o TCN, Trademark Claims Notice, Claims Notice, Trademark Notice: A Trademark Claims Notice consist of one or more Trademark Claims and are provided to prospective Registrants of DNS.
- o TCNID, Trademark Claims Notice Identifier: An element of the Trademark Claims Notice (see above), identifying said TCN. The Trademark Claims Notice Identifier is specified in the element <tmNotice:id>.
- o Trademark Claims Period: During this period, a special process applies to DNS matching the DNL List, to ensure that TMHs are informed of a DN matching their PRM. For DNS matching the DNL List, Registrars show a TCN to prospective Registrants that has to be acknowledged before effective allocation of the DN.
- o UTC: Coordinated Universal Time, as maintained by the Bureau International des Poids et Mesures (BIPM); see also [[RFC3339](#)].

4. Architecture

4.1. Sunrise Period

Architecture of the Sunrise Period

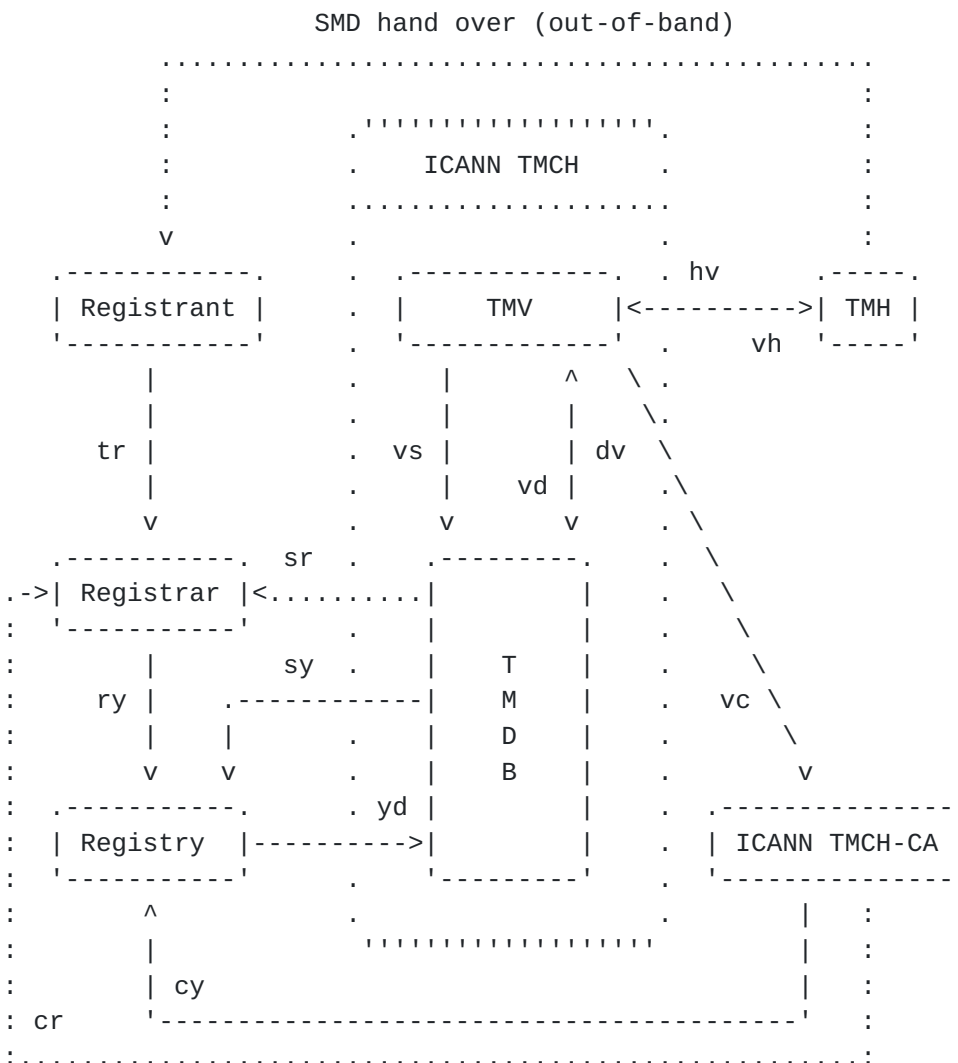


Figure 1

4.2. Trademark Claims Period

Architecture of the Trademark Claims Period

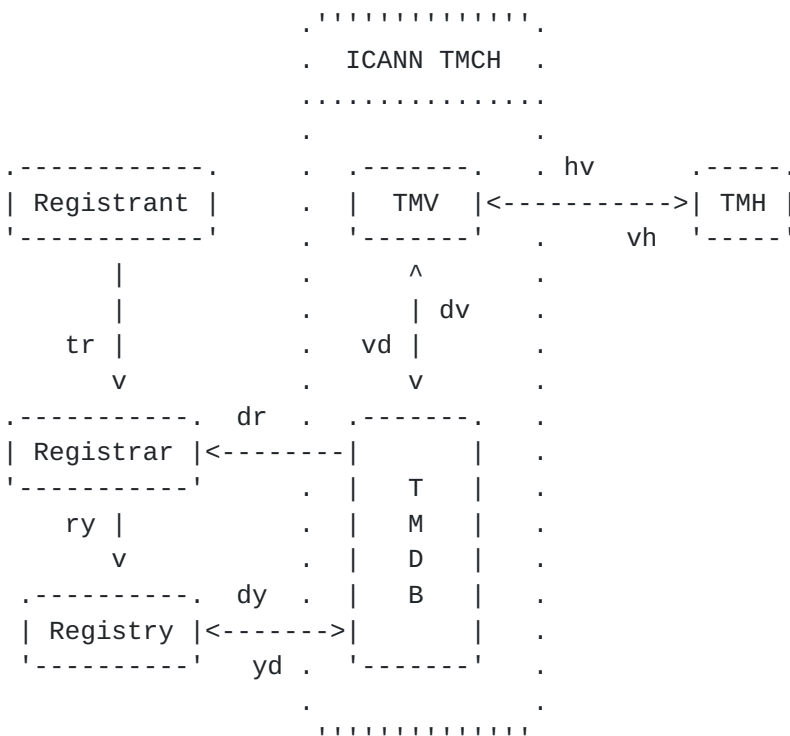


Figure 2

4.3. Interfaces

In the sub-sections below follows a short description of each interface to provide an overview of the architecture. More detailed descriptions of the relevant interfaces follow further below ([Section 5](#)).

4.3.1. hv

The TMH registers a mark with a TMV via the hv interface.

After the successful registration of the mark, the TMV makes available a SMD File (see also [Section 6.4](#)) to the TMH to be used during the Sunrise Period.

The specifics of the hv interface are beyond the scope of this document.

[4.3.2.](#) **vd**

After successful mark registration, the TMV ensures the TMDB inserts the corresponding DNLs and mark information into the database via the vd interface.

The specifics of the vd interface are beyond the scope of this document.

[4.3.3.](#) **dy**

During the Trademark Claims Period the Registry fetches the latest DNL List from the TMDB via the dy interface at regular intervals. The protocol used on the dy interface is HTTPS.

Not relevant during the Sunrise Period.

[4.3.4.](#) **tr**

The Registrant communicates with the Registrar via the tr interface.

The specifics of the tr interface are beyond the scope of this document.

[4.3.5.](#) **ry**

The Registrar communicate with the Registry via the ry interface. The ry interfaces is typically implemented in EPP.

[4.3.6.](#) **dr**

During the Trademark Claims Period, the Registrar fetches the TCN from the TMDB (to be displayed to the Registrant via the tr interface) via the dr interface. The protocol used for fetching the TCN is HTTPS.

Not relevant during the Sunrise Period.

[4.3.7.](#) **yd**

During the Sunrise Period the Registry notifies the TMDB via the yd interface of all DNS effectively allocated.

During the Trademark Claims Period, the Registry notifies the TMDB via the yd interface of all DNS effectively allocated that matched an entry in the Registry previously downloaded DNL List during the creation of the DN.

The protocol used on the yd interface is HTTPS.

[4.3.8.](#) dv

The TMDB notifies via the dv interface to the TMV of all DNS effectively allocated that match a mark registered by that TMV.

The specifics of the dv interface are beyond the scope of this document.

[4.3.9.](#) vh

The TMV notifies the TMH via the vh interface after a DN has been effectively allocated that matches a PRM of this TMH.

The specifics of the vh interface are beyond the scope of this document.

[4.3.10.](#) vs

The TMV requests to add a revoked SMD to the SMD Revocation List at the TMDB.

The specifics of the vs interface are beyond the scope of this document.

Not relevant during the Trademark Claims Period.

[4.3.11.](#) sy

During the Sunrise Period the Registry fetches the most recent SMD Revocation List from the TMDB via the sy interface in regular intervals. The protocol used on the sy interface is HTTPS.

Not relevant during the Trademark Claims Period.

[4.3.12.](#) sr

During the Sunrise Period the Registrar may fetch the most recent SMD Revocation List from the TMDB via the sr interface. The protocol used on the sr interface is the same as on the sy interface (s. above), i.e. HTTPS.

Not relevant during the Trademark Claims Period.

4.3.13. vc

The TMV registers its public key, and requests to revoke an existing key, with the ICANN TMCH-CA over the vc interface.

The specifics of the vc interface are beyond the scope of this document, but it involves personal communication between the operators of the TMV and the operators of the ICANN TMCH-CA.

Not relevant during the Trademark Claims Period.

4.3.14. cy

During the Sunrise Period the Registry fetches the most recent TMV CRL file from the ICANN TMCH-CA via the cy interface at regular intervals. The TMV CRL is used for validation of TMV certificates. The protocol used on the cy interface is HTTPS.

Not relevant during the Trademark Claims Period.

4.3.15. cr

During the Sunrise Period the Registrar optionally fetches the most recent TMV CRL file from the ICANN TMCH-CA via the cr interface at regular intervals. The TMV CRL is used for validation of TMV certificates. The protocol used on the cr interface is HTTPS.

Not relevant during the Trademark Claims Period.

5. Process Descriptions

5.1. Bootstrapping

5.1.1. Bootstrapping for Registries

5.1.1.1. Credentials

Each Registry Operator will receive authentication credentials from the TMDB to be used:

- o During the Sunrise Period to fetch the SMD Revocation List from the TMDB via the sy interface ([Section 4.3.11](#)).
- o During the Trademark Claims Period to fetch the DNL List from the TMDB via the dy interface ([Section 4.3.3](#)).
- o During the NORDN process to notify the LORDN to the TMDB via the yd interface ([Section 4.3.7](#)).

Note: credentials are created per TLD and provided to the Registry Operator.

5.1.1.2. IP Addresses for Access Control

Each Registry Operator MUST provide to the TMDB all IP addresses that will be used to:

- o Fetch the SMD Revocation List via the sy interface ([Section 4.3.11](#)).
- o Fetch the DNL List from the TMDB via the dy interface ([Section 4.3.3](#)).
- o Upload the LORDN to the TMDB via the yd interface ([Section 4.3.7](#)).

This access restriction MAY be applied by the TMDB in addition to HTTP Basic access authentication (see [[RFC7235](#)]). For credentials to be used, see [Section 5.1.1.1](#).

The TMDB MAY limit the number of IP addresses to be accepted per Registry Operator.

5.1.1.3. ICANN TMCH Trust Anchor

Each Registry Operator MUST fetch the PKIX certificate ([[RFC5280](#)]) of the ICANN TMCH-CA (Trust Anchor) from < <https://ca.icann.org/tmch.crt> > to be used:

- o During the Sunrise Period to validate the TMV certificates and the TMV CRL.

5.1.1.4. TMDB PGP Key

The TMDB MUST provide each Registry Operator with the public portion of the PGP Key used by TMDB, to be used:

- o During the Sunrise Period to perform integrity checking of the SMD Revocation List fetched from the TMDB via the sy interface ([Section 4.3.11](#)).
- o During the Trademark Claims Period to perform integrity checking of the DNL List fetched from the TMDB via the dy interface ([Section 4.3.3](#)).

5.1.2. Bootstrapping for Registrars

5.1.2.1. Credentials

Each ICANN-accredited Registrar will receive authentication credentials from the TMDB to be used:

- o During the Sunrise Period to (optionally) fetch the SMD Revocation List from the TMDB via the sr interface ([Section 4.3.12](#)).
- o During the Trademark Claims Period to fetch TCNs from the TMDB via the dr interface ([Section 4.3.6](#)).

5.1.2.2. IP Addresses for Access Control

Each Registrar MUST provide to the TMDB all IP addresses, which will be used to:

- o Fetch the SMD Revocation List via the sr interface ([Section 4.3.12](#)).
- o Fetch TCNs via the dr interface ([Section 4.3.6](#)).

This access restriction MAY be applied by the TMDB in addition to HTTP Basic access authentication (for credentials to be used, see [Section 5.1.2.1](#)).

The TMDB MAY limit the number of IP addresses to be accepted per Registrar.

5.1.2.3. ICANN TMCH Trust Anchor

Registrars MAY fetch the PKIX certificate of the ICANN TMCH-CA (Trust Anchor) from < <https://ca.icann.org/tmch.crt> > to be used:

- o During the Sunrise Period to (optionally) validate the TMV certificates and TMV CRL.

5.1.2.4. TMDB PGP Key

Registrars MUST receive the public portion of the PGP Key used by TMDB from the TMDB administrator to be used:

- o During the Sunrise Period to (optionally) perform integrity checking of the SMD Revocation List fetched from the TMDB via the sr interface ([Section 4.3.12](#)).

5.2. Sunrise Period

5.2.1. Domain Name registration

Domain Name registration during the Sunrise Period

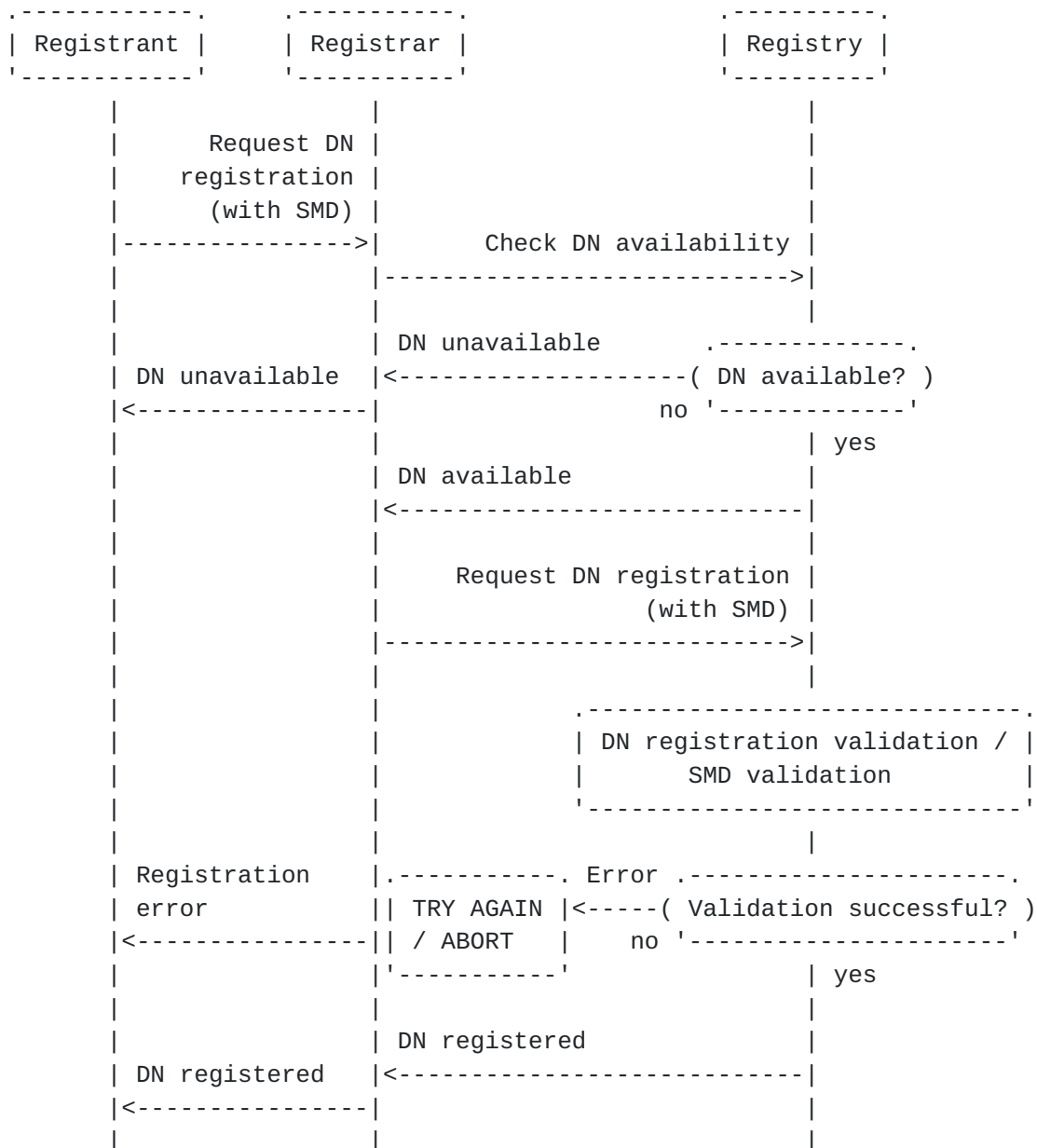


Figure 3

Note: the figure depicted above represents a synchronous DN registration workflow (usually called first come first served).

5.2.2. Sunrise Domain Name registration by Registries

Registries MUST perform a minimum set of checks for verifying each DN registration during the Sunrise Period upon reception of a registration request over the ry interface ([Section 4.3.5](#)). If any of these checks fails the Registry MUST abort the registration. Each of these checks MUST be performed before the DN is effectively allocated.

In case of asynchronous registrations (e.g. auctions), the minimum set of checks MAY be performed when creating the intermediate object (e.g. a DN application) used for DN registration. If the minimum set of checks is performed when creating the intermediate object (e.g. a DN application) a Registry MAY effectively allocate the DN without performing the minimum set of checks again.

Performing the minimum set of checks Registries MUST verify that:

1. An SMD has been received from the Registrar along with the DN registration.
2. The certificate of the TMV has been correctly signed by the ICANN TMCH-CA. (The certificate of the TMV is contained within the SMD.)
3. The datetime when the validation is done is within the validity period of the TMV certificate.
4. The certificate of the TMV is not listed in the TMV CRL file specified in the CRL distribution point of the TMV certificate.
5. The signature of the SMD (signed with the TMV certificate) is valid.
6. The datetime when the validation is done is within the validity period of the SMD based on <smd:notBefore> and <smd:notAfter> elements.
7. The SMD has not been revoked, i.e., is not contained in the SMD Revocation List.
8. The leftmost DNL of the DN being effectively allocated matches one of the labels (<mark:label>) elements in the SMD. For example, if the DN "xn--mgbachtv.xn--mgbh0fb" is being effectively allocated, the leftmost DNL would be "xn--mgbachtv".

These procedure apply to all DN effective allocations at the second level as well as to all other levels subordinate to the TLD that the Registry accepts registrations for.

5.2.3. TMDB Sunrise Services for Registries

5.2.3.1. SMD Revocation List

A new SMD Revocation List MUST be published by the TMDB twice a day, by 00:00:00 and 12:00:00 UTC.

Registries MUST refresh the latest version of the SMD Revocation List at least once every 24 hours.

Note: the SMD Revocation List will be the same regardless of the TLD. If a Backend Registry Operator manages the infrastructure of several TLDs, the Backend Registry Operator could refresh the SMD Revocation List once every 24 hours, the SMD Revocation List could be used for all the TLDs managed by the Backend Registry Operator.

Update of the SMD Revocation List



Figure 4

5.2.3.2. TMV Certificate Revocation List (CRL)

Registries MUST refresh their local copy of the TMV CRL file at least every 24 hours using the CRL distribution point specified in the TMV certificate.

Operationally, the TMV CRL file and CRL distribution point is the same for all TMVs and (at publication of this document) located at < <http://crl.icann.org/tmch.crl> >.

Note: the TMV CRL file will be the same regardless of the TLD. If a Backend Registry Operator manages the infrastructure of several TLDs, the Backend Registry Operator could refresh the TMV CRL file once every 24 hours, the TMV CRL file could be used for all the TLDs managed by the Backend Registry Operator.

Update of the TMV CRL file



Figure 5

5.2.3.3. Notice of Registered Domain Names (NORN)

The Registry MUST send a LORDN file containing DNSs effectively allocated to the TMDB (over the yd interface, [Section 4.3.7](#)).

The effective allocation of a DN MUST be reported by the Registry to the TMDB within 26 hours of the effective allocation of such DN.

The Registry MUST create and upload a LORDN file in case there are effective allocations in the SRS, that have not been successfully reported to the TMDB in a previous LORDN file.

Based on the timers used by TMVs and the TMDB, the RECOMMENDED maximum frequency to upload LORDN files from the Registries to the TMDB is every 3 hours.

It is RECOMMENDED that Registries try to upload at least two LORDN files per day to the TMDB with enough time in between, in order to have time to fix problems reported in the LORDN file.

The Registry SHOULD upload a LORDN file only when the previous LORDN file has been processed by the TMDB and the related LORDN Log file has been downloaded and processed by the Registry.

The Registry MUST upload LORDN files for DNSs effectively allocated during the Sunrise or Trademark Claims Period (same applies to DNSs effectively allocated using applications created during the Sunrise or Trademark Claims Period in case of using asynchronous registrations).

The yd interface ([Section 4.3.7](#)) MUST support at least one (1) and MAY support up to ten (10) concurrent connections from each IP address registered by a Registry Operator to access the service.

The TMDB MUST process each uploaded LORDN file and make the related log file available for Registry download within 30 minutes of the finalization of the upload.

Notification of Registered Domain Name

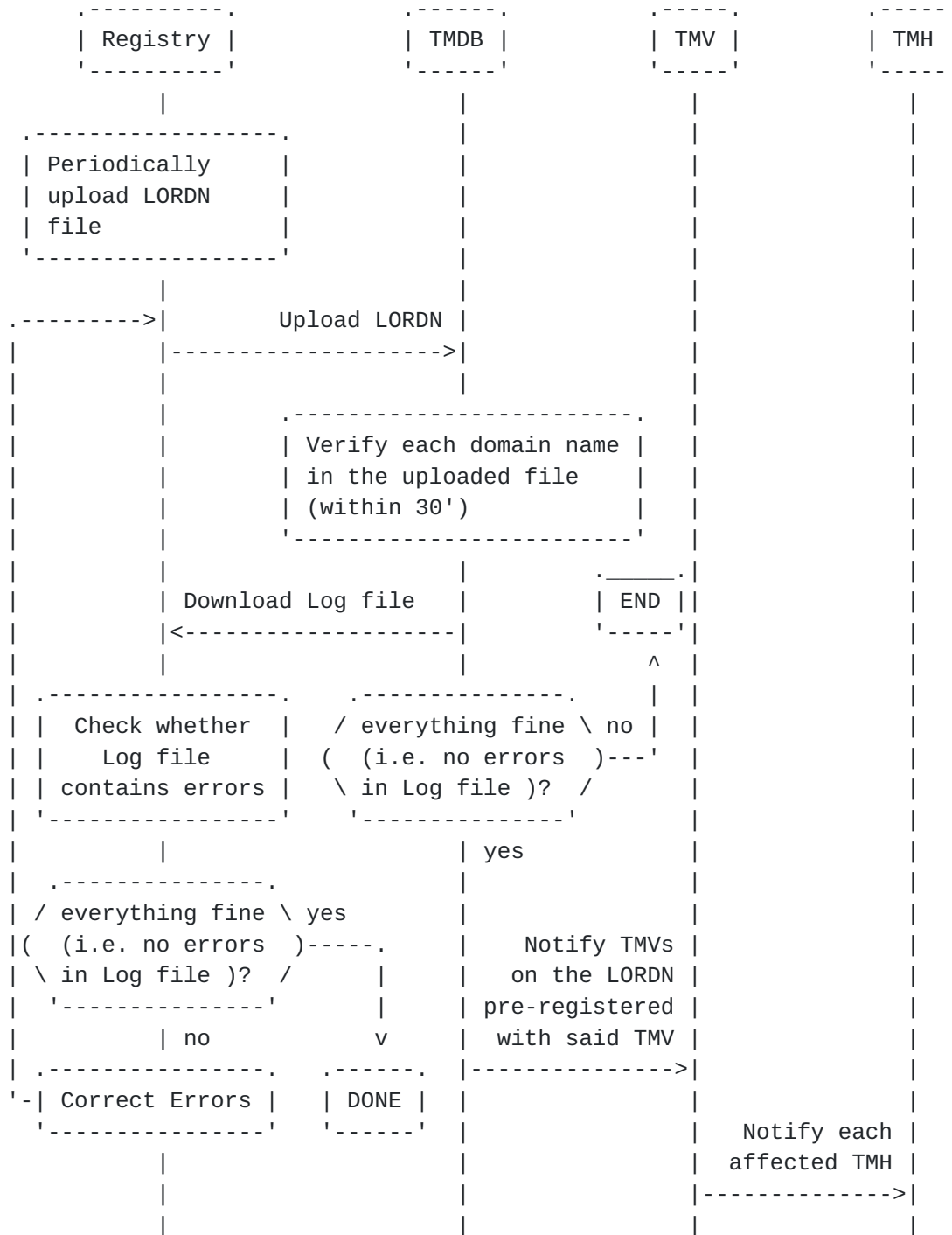


Figure 6

The format used for the LORDN is described in [Section 6.3](#)

5.2.4. Sunrise Domain Name registration by Registrars

Registrars MAY choose to perform the checks for verifying DN registrations as performed by the Registries (see [Section 5.2.2](#)) before sending the command to register a DN.

5.2.5. TMDB Sunrise Services for Registrars

The processes described in [Section 5.2.3.1](#) and [Section 5.2.3.2](#) are also available for Registrars to optionally validate the SMDs received.

5.3. Trademark Claims Period

5.3.1. Domain Registration

Domain Name registration during the Trademark Claims Period

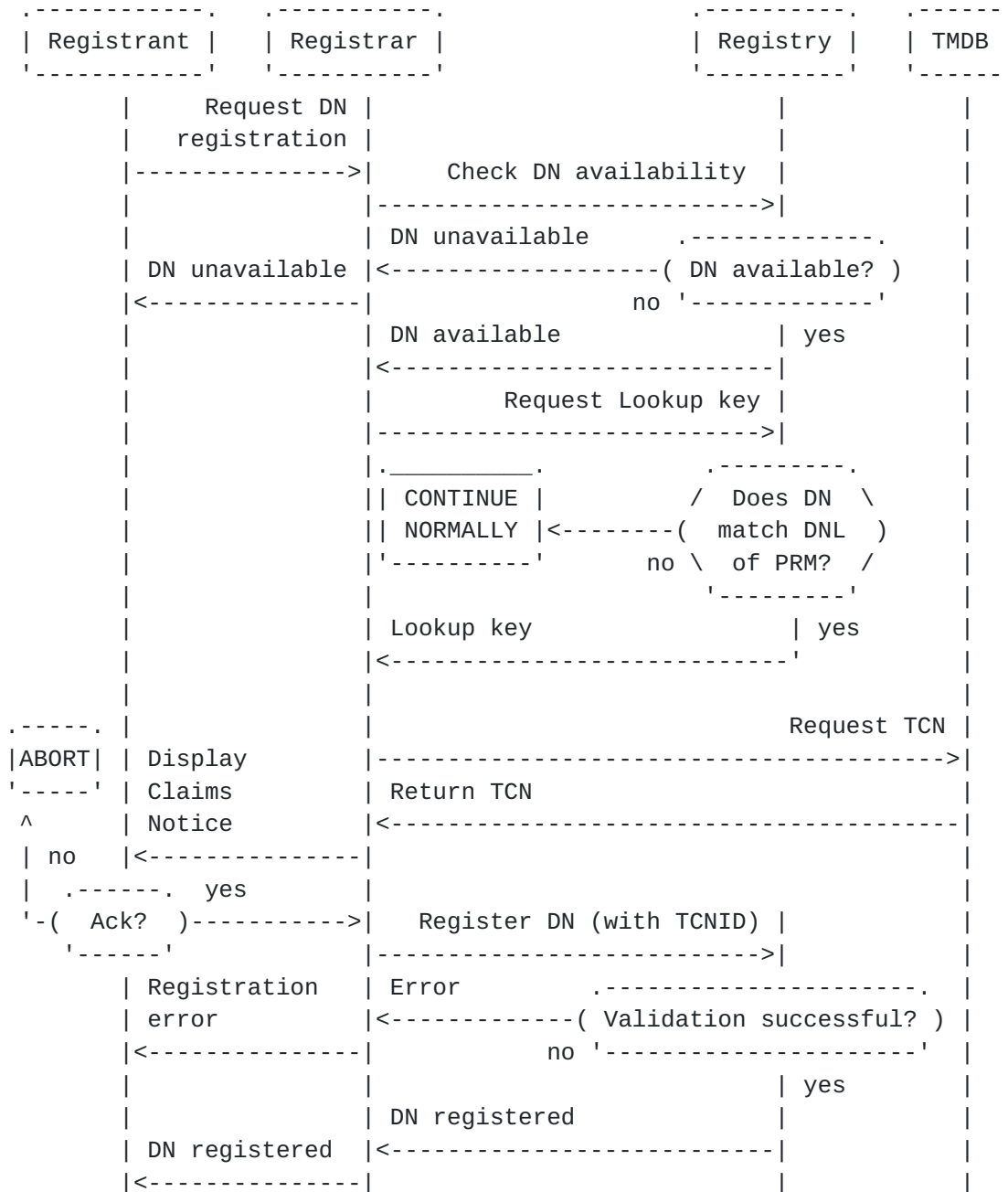


Figure 7

Note: the figure depicted above represents a synchronous DN registration workflow (usually called first come first served).

5.3.2. Trademark Claims Domain Name registration by Registries

During the Trademark Claims Period, Registries perform two main functions:

- o Registries MUST provide Registrars (over the ry interface, [Section 4.3.5](#)) the Lookup Key used to retrieve the TCNs for DNs that match the DNL List.
- o Registries MUST provide the Lookup Key only when queried about a specific DN.
- o For each DN matching a DNL of a PRM, Registries MUST perform a minimum set of checks for verifying DN registrations during the Trademark Claims Period upon reception of a registration request over the ry interface ([Section 4.3.5](#)). If any of these checks fails the Registry MUST abort the registration. Each of these checks MUST be performed before the DN is effectively allocated.
- o In case of asynchronous registrations (e.g. auctions), the minimum set of checks MAY be performed when creating the intermediate object (e.g. a DN application) used for DN effective allocation. If the minimum set of checks is performed when creating the intermediate object (e.g. a DN application) a Registry MAY effectively allocate the DN without performing the minimum set of checks again.
- o Performing the minimum set of checks Registries MUST verify that:
 1. The TCNID (<tmNotice:id>), expiration datetime (<tmNotice:notAfter>) and acceptance datetime of the TCN, have been received from the Registrar along with the DN registration.

If the three elements mentioned above are not provided by the Registrar for a DN matching a DNL of a PRM, but the DNL was inserted (or re-inserted) for the first time into DNL List less than 24 hours ago, the registration MAY continue without this data and the tests listed below are not required to be performed.

2. The TCN has not expired (according to the expiration datetime sent by the Registrar).

3. The acceptance datetime is within the window of time defined by ICANN policy. In the gTLD round of 2012, Registrars verified that the acceptance datetime was less than or equal to 48 hours in the past, as there were no defined ICANN policies at that time. Implementers should be aware that ICANN policy may define this value in the future.
4. Using the leftmost DNL of the DN being effectively allocated, the expiration datetime provided by the Registrar, and the TMDB Notice Identifier extracted from the TCNID provided by the Registrar compute the TCN Checksum. Verify that the computed TCN Checksum match the TCN Checksum present in the TCNID. For example, if the DN "xn--mgbachtv.xn--mgbh0fb" is being effectively allocated, the leftmost DNL would be "xn--mgbachtv".

These procedures apply to all DN registrations at the second level as well as to all other levels subordinate to the TLD that the Registry accepts registrations for.

5.3.3. TMDB Trademark Claims Services for Registries

5.3.3.1. Domain Name Label (DNL) List

A new DNL List MUST be published by the TMDB twice a day, by 00:00:00 and 12:00:00 UTC.

Registries MUST refresh the latest version of the DNL List at least once every 24 hours.

Update of the DNL List

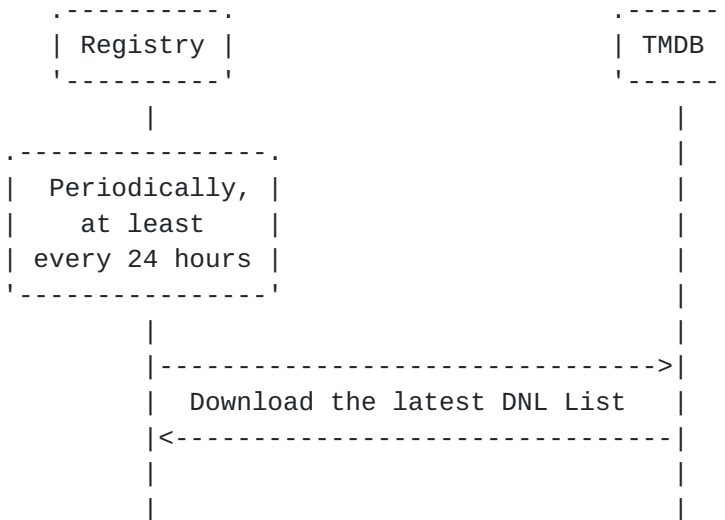


Figure 8

Note: the DNL List will be the same regardless of the TLD. If a Backend Registry Operator manages the infrastructure of several TLDs, the Backend Registry Operator could refresh the DNL List once every 24 hours, the DNL List could be used for all the TLDs managed by the Backend Registry Operator.

5.3.3.2. Notice of Registered Domain Names (NORN)

The NORDN process during the Trademark Claims Period is almost the same as during Sunrise Period as defined in [Section 5.2.3.3](#) with the difference that only registrations subject to a Trademark Claim (i.e., at registration time the name appeared in the current DNL List downloaded by the Registry Operator) are included in the LORDN.

5.3.4. Trademark Claims Domain Name registration by Registrars

For each DN matching a DNL of a PRM, Registrars MUST perform the following steps:

1. Use the Lookup Key received from the Registry to obtain the TCN from the TMDB using the dr interface ([Section 4.3.6](#)) Registrars MUST only query for the Lookup Key of a DN that is available for registration.
2. Present the TCN to the Registrant as described in Exhibit A, [[RPM-Requirements](#)].

3. Ask Registrant for acknowledgement, i.e. the Registrant MUST consent with the TCN, before any further processing. (The transmission of a TCNID to the Registry over the ry interface, [Section 4.3.5](#) implies that the Registrant has expressed his/her consent with the TCN.)
4. Perform the minimum set of checks for verifying DN registrations. If any of these checks fails the Registrar MUST abort the DN registration. Each of these checks MUST be performed before the registration is sent to the Registry. Performing the minimum set of checks Registrars MUST verify that:
 1. The datetime when the validation is done is within the TCN validity based on the <tmNotice:notBefore> and <tmNotice:notAfter> elements.
 2. The leftmost DNL of the DN being effectively allocated matches the label (<tmNotice:label>) element in the TCN. For example, if the DN "xn--mgbachtv.xn--mgbh0fb" is being effectively allocated, the leftmost DNL would be "xn--mgbachtv".
 3. The Registrant has acknowledged (expressed his/her consent with) the TCN.
5. Record the date and time when the registrant acknowledged the TCN.
6. Send the registration to the Registry (ry interface, [Section 4.3.5](#)) and include the following information:
 - * TCNID (<tmNotice:id>)
 - * Expiration date of the TCN (<tmNotice:notAfter>)
 - * Acceptance datetime of the TCN.

Currently TCNs are generated twice a day by the TMDB. The expiration date (<tmNotice:notAfter>) of each TCN MUST be set to a value defined by ICANN policy. In the gTLD round of 2012, the TMDB set the expiration value to 48 hours in to the future as there were no defined ICANN policies at that time. Implementers should be aware that ICANN policy may define this value in the future.

Registrars SHOULD implement a cache of TCNs to minimize the number of queries sent to the TMDB. A cached TCN MUST be removed from the cache after the expiration date of the TCN as defined by <tmNotice:notAfter>.

The TMDB MAY implement rate-limiting as one of the protection mechanisms to mitigate the risk of performance degradation.

[5.3.5.](#) TMDB Trademark Claims Services for Registrars

[5.3.5.1.](#) Claims Notice Information Service (CNIS)

The TCNs are provided by the TMDB online and are fetched by the Registrar via the dr interface ([Section 4.3.6](#)).

To get access to the TCNs, the Registrar needs the credentials provided by the TMDB ([Section 5.1.2.1](#)) and the Lookup Key received from the Registry via the ry interface ([Section 4.3.5](#)). The dr interface ([Section 4.3.6](#)) uses HTTPS with Basic access authentication.

The dr interface ([Section 4.3.6](#)) MAY support up to ten (10) concurrent connections from each Registrar.

The URL of the dr interface ([Section 4.3.6](#)) is:

```
< https://<tmdb-domain-name>/cnis/<lookupkey>.xml >
```

Note that the "lookupkey" may contain SLASH characters ("/"). The SLASH character is part of the URL path and MUST NOT be escaped when requesting the TCN.

The TLS certificate (HTTPS) used on the dr interface ([Section 4.3.6](#)) MUST be signed by a well-know public CA. Registrars MUST perform the Certification Path Validation described in [Section 6 of \[RFC5280\]](#). Registrars will be authenticated in the dr interface using HTTP Basic access authentication. The dr ([Section 4.3.6](#)) interface MUST support HTTPS keep-alive and MUST maintain the connection for up to 30 minutes.

[5.4.](#) Qualified Launch Program (QLP) Period

[5.4.1.](#) Domain Registration

During the OPTIONAL (see [[QLP-Addendum](#)]) Qualified Launch Program (QLP) Period effective allocations of DNs to third parties could require that Registries and Registrars provide Sunrise and/or Trademark Claims services. If required, Registries and Registrars MUST provide Sunrise and/or Trademark Claims services as described in [Section 5.2](#) and [Section 5.3](#).

The effective allocation scenarios are:

- o If the leftmost DNL of the DN being effectively allocated (QLP Name in this section) matches a DNL in the SURL, and an SMD is provided, then Registries MUST provide Sunrise Services (see [Section 5.2](#)) and the DN MUST be reported in a Sunrise LORDN file during the QLP Period. For example, if the DN "xn--mgbachtv.xn--mgbh0fb" is being effectively allocated, the leftmost DNL would be "xn--mgbachtv".
- o If the QLP Name matches a DNL in the SURL but does not match a DNL in the DNL List, and an SMD is NOT provided (see section 2.2 of [\[QLP-Addendum\]](#)), then the DN MUST be reported in a Sunrise LORDN file using the special SMD-id "99999-99999" during the QLP Period.
- o If the QLP Name matches a DNL in the SURL and also matches a DNL in the DNL List, and an SMD is NOT provided (see section 2.2 of [\[QLP-Addendum\]](#)), then Registries MUST provide Trademark Claims services (see [Section 5.3](#)) and the DN MUST be reported in a Trademark Claims LORDN file during the QLP Period.
- o If the QLP Name matches a DNL in the DNL List but does not match a DNL in the SURL, then Registries MUST provide Trademark Claims services (see [Section 5.2](#)) and the DN MUST be reported in a Trademark Claims LORDN file during the QLP Period.

The following table lists all the effective allocation scenarios during a QLP Period:

QLP Name match in the SURL List	QLP Name match in the DNL List	SMD was provided by the potential Registrant	Registry MUST provide Sunrise or Trademark Claims Services	Registry MUST report DN registration in <type> LORDN file
Y	Y	Y	Sunrise	Sunrise
Y	N	Y	Sunrise	Sunrise
N	Y	--	Trademark Claims	Trademark Claims
N	N	--	--	--
Y	Y	N (see section 2.2 of [QLP-Addendum])	Trademark Claims	Trademark Claims
Y	N	N (see section 2.2 of [QLP-Addendum])	--	Sunrise (using special SMD- id)

QLP Effective Allocation Scenarios

The TMDB MUST provide the following services to Registries during a QLP Period:

- o SMD Revocation List (see [Section 5.2.3.1](#))
- o NORN (see [Section 5.2.3.3](#))
- o DNL List (see [Section 5.3.3.1](#))
- o Sunrise List (SURL) (see [Section 5.4.2.1](#))

The TMDB MUST provide the following services to Registrars during a QLP Period:

- o SMD Revocation List (see [Section 5.2.3.1](#))

- o CNIS (see [Section 5.3.5.1](#))

5.4.2. TMDB QLP Services for Registries

5.4.2.1. Sunrise List (SURL)

A new Sunrise List (SURL) MUST be published by the TMDB twice a day, by 00:00:00 and 12:00:00 UTC.

Registries offering the OPTIONAL QLP Period MUST refresh the latest version of the SURL at least once every 24 hours.

Update of the SURL

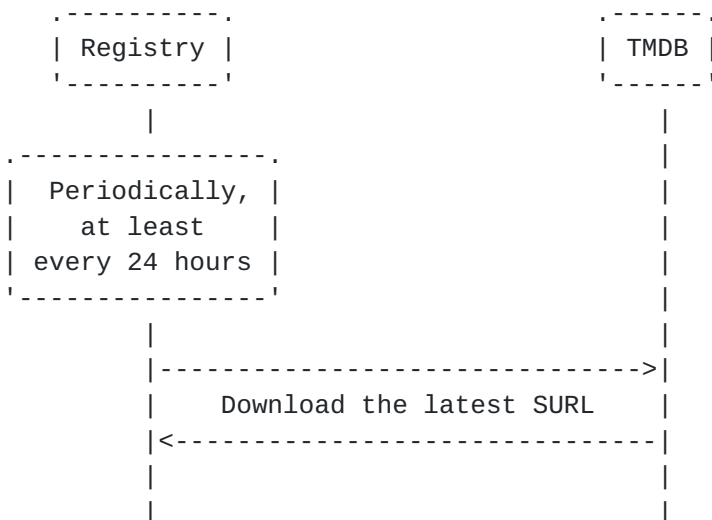


Figure 9

Note: the SURL will be the same regardless of the TLD. If a Backend Registry Operator manages the infrastructure of several TLDs, the Backend Registry Operator could refresh the SURL once every 24 hours, the SURL could be used for all the TLDs managed by the Backend Registry Operator.

6. Data Format Descriptions

6.1. Domain Name Label (DNL) List

This section defines the format of the list containing every Domain Name Label (DNL) that matches a Pre-Registered Mark (PRM). The list is maintained by the TMDB and downloaded by Registries in regular intervals (see [Section 5.3.3.1](#)). The Registries use the DNL List

during the Trademark Claims Period to check whether a requested DN matches a DNL of a PRM.

The DNL List contains all the DNLs covered by a PRM present in the TMDB at the datetime it is generated.

The DNL List is contained in a CSV formatted file that has the following structure:

- o first line: <version>,<DNL List creation datetime>

Where:

- + <version>, version of the file, this field MUST be 1.
- + <DNL List creation datetime>, date and time in UTC that the DNL List was created.

- o second line: a header line as specified in [[RFC4180](#)]

With the header names as follows:

DNL,lookup-key,insertion-datetime

- o One or more lines with: <DNL>,<lookup key>,<DNL insertion datetime>

Where:

- + <DNL>, a Domain Name Label covered by a PRM.
- + <lookup key>, lookup key that the Registry MUST provide to the Registrar. The lookup key has the following format:
<YYYY><MM><DD><vv>/<X>/<X>/<X>/<Random bits><Sequential number>, where:
 - YYYY: year that the TCN was generated.
 - MM: zero-padded month that the TCN was generated.
 - DD: zero-padded day that the TCN was generated.
 - vv: version of the TCN, possible values are 00 and 01.
 - X: one hexadecimal digit [0-9A-F]. This is the first, second and third hexadecimal digit of encoding the <Random bits> in base16 as specified in [[RFC4648](#)].

- Random bits: 144 random bits encoded in base64url as specified in [\[RFC4648\]](#).
 - Sequential number: zero-padded natural number in the range 0000000001 to 2147483647.
- + <DNL insertion datetime>, datetime in UTC that the DNL was first inserted into the DNL List. The possible two values of time for inserting a DNL to the DNL List are 00:00:00 and 12:00:00 UTC.

Example of a DNL List

```
1,2012-08-16T00:00:00.0Z
DNL,lookup-key,insertion-datetime
example,2013041500/2/6/9/rJ1NrD092vDsAzf7EQzgjX4R0000000001,\
  2010-07-14T00:00:00.0Z
another-example,2013041500/6/A/5/a1JAqG2vI2BmCv5PfUvuDkf40000000002,\
  2012-08-16T00:00:00.0Z
anotherexample,2013041500/A/C/7/rHdC4wnrWRvPY6nneCVtQhFj0000000003,\
  2011-08-16T12:00:00.0Z
```

Figure 10

To provide authentication and integrity protection, the DNL List will be PGP [\[RFC4880\]](#) signed by the TMDB (see also [Section 5.1.1.4](#)). The PGP signature of the DNL List can be found in the similar URI but with extension .sig as shown below.

The URL of the dy interface ([Section 4.3.3](#)) is:

- o < https://<tmdb-domain-name>/dnl/dnl-latest.csv >
- o < https://<tmdb-domain-name>/dnl/dnl-latest.sig >

[6.2](#). SMD Revocation List

This section defines the format of the list of SMDs that have been revoked. The list is maintained by the TMDB and downloaded by Registries (and optionally by Registrars) in regular intervals (see [Section 5.2.3.1](#)). The SMD Revocation List is used during the Sunrise Period to validate SMDs received. The SMD Revocation List has a similar function as CRLs used in PKI [\[RFC5280\]](#).

The SMD Revocation List contains all the revoked SMDs present in the TMDB at the datetime it is generated.

The SMD Revocation List is contained in a CSV formatted file that has the following structure:

- o first line: <version>,<SMD Revocation List creation datetime>

Where:

- + <version>, version of the file, this field MUST be 1.
- + <SMD Revocation List creation datetime>, datetime in UTC that the SMD Revocation List was created.

- o second line: a header line as specified in [[RFC4180](#)]

With the header names as follows:

smd-id,insertion-datetime

- o One or more lines with: <smd-id>,<revoked SMD datetime>

Where:

- + <smd-id>, identifier of the SMD that was revoked.
- + <revoked SMD datetime>, revocation datetime in UTC of the SMD. The possible two values of time for inserting an SMD to the SMD Revocation List are 00:00:00 and 12:00:00 UTC.

To provide integrity protection, the SMD Revocation List is PGP signed by the TMDB (see also [Section 5.1.1.4](#)). The SMD Revocation List is provided by the TMDB with extension .csv. The PGP signature of the SMD Revocation List can be found in the similar URI but with extension .sig as shown below.

The URL of the sr interface ([Section 4.3.12](#)) and sy interface ([Section 4.3.11](#)) is:

- o < https://<tmdb-domain-name>/smdrl/smdrl-latest.csv >
- o < https://<tmdb-domain-name>/smdrl/smdrl-latest.sig >

Example of an SMD Revocation List

```
1,2012-08-16T00:00:00.0Z  
smd-id,insertion-datetime  
2-2,2012-08-15T00:00:00.0Z  
3-2,2012-08-15T00:00:00.0Z  
1-2,2012-08-15T00:00:00.0Z
```

Figure 11

6.3. List of Registered Domain Names (LORDN) file

This section defines the format of the List of Registered Domain Names (LORDN), which is maintained by each Registry and uploaded at least daily to the TMDB. Every time a DN matching a DNL of a PRM said DN is added to the LORDN along with further information related to its registration.

The URIs of the yd interface ([Section 4.3.7](#)) used to upload the LORDN file is:

- o Sunrise LORDN file:

< https://<tmdb-domain-name>/LORDN/<TLD>/sunrise >

- o Trademark Claims LORDN file:

< https://<tmdb-domain-name>/LORDN/<TLD>/claims >

During a QLP Period, Registries MAY be required to upload Sunrise or Trademark Claims LORDN files. The URIs of the yd interface used to upload LORDN files during a QLP Period is:

- o Sunrise LORDN file (during QLP Period):

< https://<tmdb-domain-name>/LORDN/<TLD>/sunrise/qlp >

- o Trademark Claims LORDN file (during a QLP Period):

< https://<tmdb-domain-name>/LORDN/<TLD>/claims/qlp >

The yd interface ([Section 4.3.7](#)) returns the following HTTP status codes after a HTTP POST request method is received:

- o The interface provides a HTTP/202 status code if the interface was able to receive the LORDN file and the syntax of the LORDN file is correct.

The interface provides the LORDN Transaction Identifier in the HTTP Entity-body that would be used by the Registry to download the LORDN Log file. The LORDN Transaction Identifier is a natural number zero-padded in the range 00000000000000000001 to 9223372036854775807.

The TMDB uses the <LORDN creation datetime> element of the LORDN file as a unique client-side identifier. If a LORDN file with the same <LORDN creation datetime> of a previously sent LORDN file is received by the TMDB, the LORDN Transaction Identifier of the previously sent LORDN file MUST be provided to the Registry. The TMDB MUST ignore the DN Lines present in the LORDN file if a LORDN file with the same <LORDN creation datetime> was previously sent.

The HTTP Location header field contains the URI where the LORDN Log file could be retrieved later, for example:

202 Accepted

Location: https://<tmdb-domain-name>/LORDN/example/sunrise/00000000000000000001/result

- o The interface provides a HTTP/400 if the request is incorrect or the syntax of the LORDN file is incorrect. The TMDB MUST return a human readable message in the HTTP Entity-body regarding the incorrect syntax of the LORDN file.
- o The interface provides a HTTP/401 status code if the credentials provided does not authorize the Registry Operator to upload a LORDN file.
- o The TMDB MUST return a HTTP/404 status code when trying to upload a LORDN file using the https://<tmdb-domain-name>/LORDN/<TLD>/sunrise/qlp or https://<tmdb-domain-name>/LORDN/<TLD>/claims/qlp interface outside of a QLP Period plus 26 hours.
- o The interface provides a HTTP/500 status code if the system is experiencing a general failure.

For example, to upload the Sunrise LORDN file for TLD "example", the URI would be:

< https://<tmdb-domain-name>/LORDN/example/sunrise >

The LORDN is contained in a CSV formatted file that has the following structure:

o For Sunrise Period:

- * first line: <version>,<LORDN creation datetime>,<Number of DN Lines>

Where:

- <version>, version of the file, this field MUST be 1.
- <LORDN creation datetime>, date and time in UTC that the LORDN was created.
- <Number of DN Lines>, number of DN Lines present in the LORDN file.

- * second line: a header line as specified in [[RFC4180](#)]

With the header names as follows:

roid, domain-name, SMD-id, registrar-id, registration-datetime, application-datetime

- * One or more lines with: <roid>,<DN registered>,<SMD-id>,<IANA Registrar id>,<datetime of registration>,<datetime of application creation>

Where:

- <roid>, DN Repository Object Identifier (DNROID) in the SRS.
- <DN registered>, DN that was effectively allocated. For IDNs, the A-label form is used.
- <SMD-id>, SMD ID used for registration.
- <IANA Registrar ID>, IANA Registrar ID.
- <datetime of registration>, date and time in UTC that the domain was effectively allocated.
- OPTIONAL <datetime of application creation>, date and time in UTC that the application was created. The

<datetime of application creation> MUST be provided in case of a DN effective allocation based on an asynchronous registration (e.g., when using auctions).

Example of a Sunrise LORDN file

```
1,2012-08-16T00:00:00.0Z,3
roid, domain-name, SMD-id, registrar-id, registration-datetime, \
  application-datetime
SH8013-REP, example1.gtld, 1-2, 9999, 2012-08-15T13:20:00.0Z, \
  2012-07-15T00:50:00.0Z
EK77-REP, example2.gtld, 2-2, 9999, 2012-08-15T14:00:03.0Z
HB800-REP, example3.gtld, 3-2, 9999, 2012-08-15T15:40:00.0Z
```

Figure 12

o For Trademark Claims Period:

- * first line: <version>, <LORDN creation datetime>, <Number of DN Lines>

Where:

- <version>, version of the file, this field MUST be 1.
- <LORDN creation datetime>, date and time in UTC that the LORDN was created.
- <Number of DN Lines>, number of DN Lines present in the LORDN file.

- * second line: a header line as specified in [[RFC4180](#)]

With the header names as follows:

```
roid, domain-name, notice-id, registrar-id, registration-
datetime, ack-datetime, application-datetime
```

- * One or more lines with: <roid>, <DN registered>, <TCNID>, <IANA Registrar id>, <datetime of registration>, <datetime of acceptance of the TCN>, <datetime of application creation>

Where:

- <roid>, DN Repository Object Identifier (DNROID) in the SRS.

- <DN registered>, DN that was effectively allocated. For IDNs, the A-label form is used.
- <TCNID>, Trademark Claims Notice Identifier as specified in <tmNotice:id>.
- <IANA Registrar ID>, IANA Registrar ID.
- <datetime of registration>, date and time in UTC that the domain was effectively allocated.
- <datetime of acceptance of the TCN>, date and time in UTC that the TCN was acknowledged.
- OPTIONAL <datetime of application creation>, date and time in UTC that the application was created. The <datetime of application creation> MUST be provided in case of a DN effective allocation based on an asynchronous registration (e.g., when using auctions).

For a DN matching a DNL of a PRM at the moment of registration, created without the TCNID, expiration datetime and acceptance datetime, because DNL was inserted (or re-inserted) for the first time into DNL List less than 24 hours ago, the string "recent-dnl-insertion" MAY be specified in <TCNID> and <datetime of acceptance of the TCN>.

Example of a Trademark Claims LORDN file

```
1,2012-08-16T00:00:00.0Z,3
roid, domain-name, notice-id, registrar-id, registration-datetime, \
  ack-datetime, application-datetime
SH8013-REP, example1.gtld, a76716ed9223352036854775808, \
  9999, 2012-08-15T14:20:00.0Z, 2012-08-15T13:20:00.0Z
EK77-REP, example2.gtld, a7b786ed9223372036856775808, \
  9999, 2012-08-15T11:20:00.0Z, 2012-08-15T11:19:00.0Z
HB800-REP, example3.gtld, recent-dnl-insertion, \
  9999, 2012-08-15T13:20:00.0Z, recent-dnl-insertion
```

Figure 13

6.3.1. LORDN Log file

After reception of the LORDN file, the TMDB verifies its content for syntactical and semantical correctness. The output of the LORDN file verification is retrieved using the yd interface ([Section 4.3.7](#)).

The URI of the yd interface ([Section 4.3.7](#)) used to retrieve the LORDN Log file is:

- o Sunrise LORDN Log file:

```
< https://<tmdb-domain-name>/LORDN/<TLD>/sunrise/<lordn-transaction-identifier>/result >
```

- o Trademark Claims LORDN Log file:

```
< https://<tmdb-domain-name>/LORDN/<TLD>/claims/<lordn-transaction-identifier>/result >
```

A Registry Operator MUST NOT send more than one request per minute per TLD to download a LORDN Log file.

The yd interface ([Section 4.3.7](#)) returns the following HTTP status codes after a HTTP GET request method is received:

- o The interface provides a HTTP/200 status code if the interface was able to provide the LORDN Log file. The LORDN Log file is contained in the HTTP Entity-body.
- o The interface provides a HTTP/204 status code if the LORDN Transaction Identifier is correct, but the server has not finalized processing the LORDN file.
- o The interface provides a HTTP/400 status code if the request is incorrect.
- o The interface provides a HTTP/401 status code if the credentials provided does not authorize the Registry Operator to download the LORDN Log file.
- o The interface provides a HTTP/404 status code if the LORDN Transaction Identifier is incorrect.
- o The interface provides a HTTP/500 status code if the system is experiencing a general failure.

For example, to obtain the LORDN Log file in case of a Sunrise LORDN file with LORDN Transaction Identifier 00000000000000000001 and TLD "example" the URI would be:

```
< https://<tmdb-domain-  
name>/LORDN/example/sunrise/00000000000000000001/result >
```

The LORDN Log file is contained in a CSV formatted file that has the following structure:

- o first line: <version>,<LORDN Log creation datetime>,<LORDN file creation datetime>,<LORDN Log Identifier>,<Status flag>,<Warning flag>,<Number of DN Lines>

Where:

- + <version>, version of the file, this field MUST be 1.
- + <LORDN Log creation datetime>, date and time in UTC that the LORDN Log was created.
- + <LORDN file creation datetime>, date and time in UTC of creation for the LORDN file that this log file is referring to.
- + <LORDN Log Identifier>, unique identifier of the LORDN Log provided by the TMDB. This identifier could be used by the Registry Operator to unequivocally identify the LORDN Log. The identifier will be a string of a maximum LENGTH of 60 characters from the Base 64 alphabet.
- + <Status flag>, whether the LORDN file has been accepted for processing by the TMDB. Possible values are "accepted" or "rejected".
- + <Warning flag>, whether the LORDN Log has any warning result codes. Possible values are "no-warnings" or "warnings-present".
- + <Number of DN Lines>, number of DNs effective allocations processed in the LORDN file.

A Registry Operator is NOT REQUIRED to process a LORDN Log with a <Status flag>="accepted" and <Warning flag>="no-warnings".

- o second line: a header line as specified in [[RFC4180](#)]

With the header names as follows:

- roid,result-code
- o One or more lines with: <roid>,<result code>
- Where:
- + <roid>, DN Repository Object IDentifier (DNROID) in the SRS.
 - + <result code>, result code as described in [Section 6.3.1.1](#).

Example of a LORDN Log file

```
1,2012-08-16T02:15:00.0Z,2012-08-16T00:00:00.0Z,\
00000000000000478Nzs+3VMkR8ckuUynOLmyeqTmZQSbzDuf/R50n2n5QX4=,\
accepted,no-warnings,1
roid,result-code
SH8013-REP,2000
```

Figure 14

[6.3.1.1](#). LORDN Log Result Codes

In Figure 15 the classes of result codes (rc) are listed. Those classes in square brackets are not used at this time, but may come into use at some later stage. The first two digits of a result code denote the result code class, which defines the outcome at the TMDB:

- o ok: Success, DN Line accepted by the TMDB.
- o warn: a warning is issued, DN Line accepted by the TMDB.
- o err: an error is issued, LORDN file rejected by the TMDB.

In case that after processing a DN Line, the error result code is 45xx or 46xx for that DN Line, the LORDN file MUST be rejected by the TMDB. If the LORDN file is rejected, DN Lines that are syntactically valid will be reported with a 2001 result code. A 2001 result code means that the DN Line is syntactically valid, however the DN Line was not processed because the LORDN file was rejected. All DNS reported in a rejected LORDN file MUST be reported again by the Registry because none of the DN Lines present in the LORDN file have been processed by the TMDB.

LORDN Log Result Code Classes

code	Class	outcome
----	-----	-----
20xx	Success	ok
35xx	[DN Line syntax warning]	warn
36xx	DN Line semantic warning	warn
45xx	DN Line syntax error	err
46xx	DN Line semantic error	err

Figure 15

In the following, the LORDN Log result codes used by the TMDB are described:

LORDN Log result Codes

rc	Short Description Long Description
----	-----
2000	OK DN Line successfully processed.
2001	OK but not processed DN Line is syntactically correct but was not processed because the LORDN file was rejected.
3601	TCN Acceptance Date after Registration Date TCN Acceptance Date in DN Line is newer than the Registration Date.
3602	Duplicate DN Line This DN Line is an exact duplicate of another DN Line in same file, DN Line ignored.
3603	DNROID Notified Earlier Same DNROID has been notified earlier, DN Line ignored.
3604	TCN Checksum invalid Based on the DN effectively allocated, the TCNID and the expiration date of the linked TCN, the TCN Checksum is

invalid.

[3605](#) TCN Expired

The TCN was already expired (based on the <tmNotice:notAfter> field of the TCN) at the datetime of acknowledgement.

[3606](#) Wrong TCNID used

The TCNID used for the registration does not match the related DN.

[3609](#) Invalid SMD used

The SMD used for registration was not valid at the moment of registration based on the <smd:notBefore> and <smd:notAfter> elements.

In case of an asynchronous registration, this refer to the <datetime of application creation>.

[3610](#) DN reported outside of the time window

The DN was reported outside of the required 26 hours reporting window.

[3611](#) DN does not match the labels in SMD

The DN does not match the labels included in the SMD.

[3612](#) SMDID does not exist

The SMDID has never existed in the central repository.

[3613](#) SMD was revoked when used

The SMD used for registration was revoked more than 24 hours ago of the <datetime of registration>.

In case of an asynchronous registration, the <datetime of application creation> is used when validating the DN Line.

[3614](#) TCNID does not exist

The TCNID has never existed in the central repository.

[3615](#) Recent-dnl-insertion outside of the time window

The DN registration is reported as a recent-dnl-insertion, but the (re) insertion into the DNL occurred more than 24 hours ago.

[3616](#) Registration Date of DN in Claims before the end of Sunrise Period

The registration date of the DN is before the end of Sunrise Period and the DN was reported in a Trademark Claims LORDN file.

[3617](#) Registrar has not been approved by the TMDB

Registrar ID in DN Line has not completed Trademark Claims integration

testing with the TMDB.

[3618](#) Registration Date of DN in a QLP LORDN file outside of the QLP Period
The registration date of the DN in a QLP LORDN file is outside of the QLP Period.

[3619](#) TCN was not valid
The TCN was not valid (based on the <tmNotice:notBefore> field of the TCN) at the datetime of acknowledgement.

[4501](#) Syntax Error in DN Line
Syntax Error in DN Line.

[4601](#) Invalid TLD used
The TLD in the DN Line does not match what is expected for this LORDN.

[4602](#) Registrar ID Invalid
Registrar ID in DN Line is not a valid ICANN-Accredited Registrar.

[4603](#) Registration Date in the future
The <datetime of registration> in the DN Line is in the future.

[4606](#) TLD not in Sunrise or Trademark Claims Period
The <datetime of registration> was reported when the TLD was not in Sunrise or Trademark Claims Periods.
In case of an asynchronous registration, the <datetime of application creation> is used when validating the DN Line.

[4607](#) Application Date in the future
The <datetime of application creation> in the DN Line is in the future.

[4608](#) Application Date is later than Registration Date
The <datetime of application creation> in the DN Line is later than the <datetime of registration>.

[4609](#) TCNID wrong syntax
The syntax of the TCNID is invalid.

[4610](#) TCN Acceptance Date is in the future
The <datetime of acceptance of the TCN> is in the future.

[4611](#) Label has never existed in the TMDB
The label in the registered DN has never existed in the TMDB.

Figure 16

6.4. Signed Mark Data (SMD) File

This section defines the format of the Signed Mark Data (SMD) File. After a successful registration of a mark, the TMV returns an SMD File to the TMH. The SMD File can then be used for registration of one or more DNS covered by the PRM during the Sunrise Period of a TLD.

Two encapsulation boundaries are defined for delimiting the encapsulated base64 encoded SMD: i.e. "-----BEGIN ENCODED SMD-----" and "-----END ENCODED SMD-----". Only data inside the encapsulation boundaries MUST be used by Registries and Registrars for validation purposes, i.e. any data outside these boundaries as well as the boundaries themselves MUST be ignored for validation purposes.

The structure of the SMD File is as follows, all the elements are REQUIRED, and MUST appear in the specified order.

1. Marks: <marks>
2. smdID: <SMD-ID>
3. U-labels: <comma separated list of U-label or NR-LDH labels (see [\[RFC5890\]](#))>
4. notBefore: <begin validity>
5. notAfter: <end validity>
6. -----BEGIN ENCODED SMD-----
7. <encoded SMD (see [\[RFC7848\]](#))>
8. -----END ENCODED SMD-----

Example of an SMD File:

```
Marks: Example One
smdID: 1-2
U-labels: example-one, exampleone
notBefore: 2011-08-16 09:00
notAfter: 2012-08-16 09:00
-----BEGIN ENCODED SMD-----
PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz4KPHNtZDpzaWdu
ZWRNYXJrIHhtbG5zOnNtZD0idXJuOmllldGY6cGFyYW1zOnhtbDpuczpzaWduZWRN
... (base64 data elided for brevity) ...
dXJlPgo8L3NtZDpzaWduZWRNYXJrPgo=
-----END ENCODED SMD-----
```

Figure 17

6.5. Trademark Claims Notice (TCN)

The TMDB MUST provide the TCN to Registrars in XML format as specified below.

An enclosing element `<tmNotice:notice>` that describes the Trademark Notice to a given label.

The child elements of the `<tmNotice:notice>` element include:

- o A `<tmNotice:id>` element that contains the unique identifier of the Trademark Notice. This element contains the the TCNID.

The TCNID is a string concatenation of a TCN Checksum and the TMDB Notice Identifier. The first 8 characters of the TCNID is a TCN Checksum. The rest is the TMDB Notice Identifier, which is a zero-padded natural number in the range of 00000000000000000001 to 9223372036854775807.

Example of a TCNID:

370d0b7c9223372036854775807.

Where:

- + TCN Checksum=370d0b7c
- + TMDB Notice Identifier=9223372036854775807

The TCN Checksum is a 8 characters long Base16 encoded output of computing the CRC32 of the string concatenation of: label + unix_timestamp(<tmNotice:notAfter>) + TMDB Notice Identifier

TMDB MUST use the Unix time conversion of the <tmNotice:notAfter> in UTC to calculate the TCN Checksum. Unix time is defined as the number of seconds that have elapsed since 1970-01-01T00:00:00Z not counting leap seconds. For example, the conversion to Unix time of 2010-08-16T09:00:00.0Z is shown:

```
unix_time(2010-08-16T09:00:00.0Z)=1281949200
```

The TMDB uses the <tmNotice:label> and <tmNotice:notAfter> elements from the TCN along with the TMDB Notice Identifier to compute the TCN Checksum.

A Registry MUST use the leftmost DNL of the DN being effectively allocated, the expiration datetime of the TCN (provided by the Registrar) and the TMDB Notice Identifier extracted from the TCNID (provided by the Registrar) to compute the TCN Checksum. For example, if the DN "xn--mgbachtv.xn--mgbh0fb" is being effectively allocated, the leftmost DNL would be "xn--mgbachtv".

Example of computation of the TCN Checksum:

```
CRC32(example-one12819492009223372036854775807)=370d0b7c
```

- o A <tmNotice:notBefore> element that contains the start of the validity date and time of the TCN.
- o A <tmNotice:notAfter> element that contains the expiration date and time of the TCN.
- o A <tmNotice:label> element that contains the DNL covered by a PRM.
- o One or more <tmNotice:claim> elements that contain the Trademark Claim. The <tmNotice:claim> element contains the following child elements:
 - * A <tmNotice:markName> element that contains the mark text string.
 - * One or more <tmNotice:holder> elements that contains the information of the holder of the mark. An "entitlement" attribute is used to identify the entitlement of the holder,

possible values are: owner, assignee or licensee. The child elements of <tmNotice:holder> include:

- + An OPTIONAL <tmNotice:name> element that contains the name of the holder. A <tmNotice:name> MUST be specified if <tmNotice:org> is not specified.
- + An OPTIONAL <tmNotice:org> element that contains the name of the organization holder of the mark. A <tmNotice:org> MUST be specified if <tmNotice:name> is not specified.
- + A <tmNotice:addr> element that contains the address information of the holder of a mark. A <tmNotice:addr> contains the following child elements:
 - One, two or three OPTIONAL <tmNotice:street> elements that contains the organization's street address.
 - A <tmNotice:city> element that contains the organization's city.
 - An OPTIONAL <tmNotice:sp> element that contains the organization's state or province.
 - An OPTIONAL <tmNotice:pc> element that contains the organization's postal code.
 - A <tmNotice:cc> element that contains the organization's country code. This a two-character code from [\[ISO3166-2\]](#).
- + An OPTIONAL <tmNotice:voice> element that contains the organization's voice telephone number.
- + An OPTIONAL <tmNotice:fax> element that contains the organization's facsimile telephone number.
- + An OPTIONAL <tmNotice:email> element that contains the email address of the holder.
- * Zero or more OPTIONAL <tmNotice:contact> elements that contains the information of the representative of the mark registration. A "type" attribute is used to identify the type of contact, possible values are: owner, agent or thirdparty. The child elements of <tmNotice:contact> include:
 - + A <tmNotice:name> element that contains name of the responsible person.

- + An OPTIONAL <tmNotice:org> element that contains the name of the organization of the contact.
- + A <tmNotice:addr> element that contains the address information of the contact. A <tmNotice:addr> contains the following child elements:
 - One, two or three OPTIONAL <tmNotice:street> elements that contains the contact's street address.
 - A <tmNotice:city> element that contains the contact's city.
 - An OPTIONAL <tmNotice:sp> element that contains the contact's state or province.
 - An OPTIONAL <tmNotice:pc> element that contains the contact's postal code.
 - A <tmNotice:cc> element that contains the contact's country code. This a two-character code from [\[ISO3166-2\]](#).
- + A <tmNotice:voice> element that contains the contact's voice telephone number.
- + An OPTIONAL <tmNotice:fax> element that contains the contact's facsimile telephone number.
- + A <tmNotice:email> element that contains the contact's email address.
- * A <tmNotice:jurDesc> element that contains the name (in English) of the jurisdiction where the mark is protected. A jurCC attribute contains the two-character code of the jurisdiction where the mark was registered. This is a two-character code from [\[WIPO.ST3\]](#).
- * Zero or more OPTIONAL <tmNotice:classDesc> element that contains the description (in English) of the Nice Classification as defined in [\[WIPO-NICE-CLASSES\]](#). A classNum attribute contains the class number.
- * A <tmNotice:goodsAndServices> element that contains the full description of the goods and services mentioned in the mark registration document.

- * An OPTIONAL `<tmNotice:notExactMatch>` element signals that the claim notice was added to the TCN based on other rule (e.g. [[Claims50](#)]) than exact match (defined in [[MatchingRules](#)]). The `<tmNotice:notExactMatch>` contains one or more:
 - + An OPTIONAL `<tmNotice:udrp>` element that signals that the claim notice was added because of a previously abused name included in an UDRP case. The `<tmNotice:udrp>` contains:
 - A `<tmNotice:caseNo>` element that contains the UDRP case number used to validate the previously abused name.
 - A `<tmNotice:udrpProvider>` element that contains the name of the UDRP provider.
 - + An OPTIONAL `<tmNotice:court>` element that signals that the claim notice was added because of a previously abused name included in a court's resolution. The `<tmNotice:court>` contains:
 - A `<tmNotice:refNum>` element that contains the reference number of the court's resolution used to validate the previously abused name.
 - A `<tmNotice:cc>` element that contains the two-character code from [[ISO3166-2](#)] of the jurisdiction of the court.
 - A `<tmNotice:courtName>` element that contains the name of the court.

Example of a `<tmNotice:notice>` object:

```
<?xml version="1.0" encoding="UTF-8"?>
<tmNotice:notice xmlns:tmNotice="urn:ietf:params:xml:ns:tmNotice-1.0">
  <tmNotice:id>370d0b7c9223372036854775807</tmNotice:id>
  <tmNotice:notBefore>2010-08-14T09:00:00.0Z</tmNotice:notBefore>
  <tmNotice:notAfter>2010-08-16T09:00:00.0Z</tmNotice:notAfter>
  <tmNotice:label>example-one</tmNotice:label>
  <tmNotice:claim>
    <tmNotice:markName>Example One</tmNotice:markName>
    <tmNotice:holder entitlement="owner">
      <tmNotice:org>Example Inc.</tmNotice:org>
      <tmNotice:addr>
        <tmNotice:street>123 Example Dr.</tmNotice:street>
        <tmNotice:street>Suite 100</tmNotice:street>
        <tmNotice:city>Reston</tmNotice:city>
        <tmNotice:sp>VA</tmNotice:sp>
```



```
<tmNotice:pc>20190</tmNotice:pc>
<tmNotice:cc>US</tmNotice:cc>
</tmNotice:addr>
</tmNotice:holder>
<tmNotice:contact type="owner">
  <tmNotice:name>Joe Doe</tmNotice:name>
  <tmNotice:org>Example Inc.</tmNotice:org>
  <tmNotice:addr>
    <tmNotice:street>123 Example Dr.</tmNotice:street>
    <tmNotice:street>Suite 100</tmNotice:street>
    <tmNotice:city>Reston</tmNotice:city>
    <tmNotice:sp>VA</tmNotice:sp>
    <tmNotice:pc>20190</tmNotice:pc>
    <tmNotice:cc>US</tmNotice:cc>
  </tmNotice:addr>
  <tmNotice:voice x="4321">+1.7035555555</tmNotice:voice>
  <tmNotice:email>jdoe@example.com</tmNotice:email>
</tmNotice:contact>
<tmNotice:jurDesc jurCC="US">UNITED STATES OF AMERICA</tmNotice:jurDesc>
<tmNotice:classDesc classNum="35">
  Advertising; business management; business administration.
</tmNotice:classDesc>
<tmNotice:classDesc classNum="36">
  Insurance; financial affairs; monetary affairs; real estate.
</tmNotice:classDesc>
<tmNotice:goodsAndServices>
  Bardus populorum circumdabit se cum captiosus populum.
  Smert populorum circumdabit se cum captiosus populum qui eis differimus.
</tmNotice:goodsAndServices>
</tmNotice:claim>
<tmNotice:claim>
  <tmNotice:markName>Example-One</tmNotice:markName>
  <tmNotice:holder entitlement="owner">
    <tmNotice:org>Example S.A. de C.V.</tmNotice:org>
    <tmNotice:addr>
      <tmNotice:street>Calle conocida #343</tmNotice:street>
      <tmNotice:city>Conocida</tmNotice:city>
      <tmNotice:sp>SP</tmNotice:sp>
      <tmNotice:pc>82140</tmNotice:pc>
      <tmNotice:cc>BR</tmNotice:cc>
    </tmNotice:addr>
  </tmNotice:holder>
  <tmNotice:jurDesc jurCC="BR">BRAZIL</tmNotice:jurDesc>
  <tmNotice:goodsAndServices>
    Bardus populorum circumdabit se cum captiosus populum.
    Smert populorum circumdabit se cum captiosus populum qui eis differimus.
  </tmNotice:goodsAndServices>
</tmNotice:claim>
```



```
<tmNotice:claim>
  <tmNotice:markName>One</tmNotice:markName>
  <tmNotice:holder entitlement="owner">
    <tmNotice:org>One Corporation</tmNotice:org>
    <tmNotice:addr>
      <tmNotice:street>Otra calle</tmNotice:street>
      <tmNotice:city>Otra ciudad</tmNotice:city>
      <tmNotice:sp>OT</tmNotice:sp>
      <tmNotice:pc>383742</tmNotice:pc>
      <tmNotice:cc>CR</tmNotice:cc>
    </tmNotice:addr>
  </tmNotice:holder>
  <tmNotice:jurDesc jurCC="CR">COSTA RICA</tmNotice:jurDesc>
  <tmNotice:goodsAndServices>
    Bardus populum circumdabit se cum captiosus populum.
    Smert populum circumdabit se cum captiosus populum qui eis differimus.
  </tmNotice:goodsAndServices>
  <tmNotice:notExactMatch>
    <tmNotice:court>
      <tmNotice:refNum>234235</tmNotice:refNum>
      <tmNotice:cc>CR</tmNotice:cc>
      <tmNotice:courtName>Supreme Court of Justice of Costa Rica</
tmNotice:courtName>
    </tmNotice:court>
  </tmNotice:notExactMatch>
</tmNotice:claim>
<tmNotice:claim>
  <tmNotice:markName>One Inc</tmNotice:markName>
  <tmNotice:holder entitlement="owner">
    <tmNotice:org>One SA de CV</tmNotice:org>
    <tmNotice:addr>
      <tmNotice:street>La calle</tmNotice:street>
      <tmNotice:city>La ciudad</tmNotice:city>
      <tmNotice:sp>CD</tmNotice:sp>
      <tmNotice:pc>34323</tmNotice:pc>
      <tmNotice:cc>AR</tmNotice:cc>
    </tmNotice:addr>
  </tmNotice:holder>
  <tmNotice:jurDesc jurCC="AR">ARGENTINA</tmNotice:jurDesc>
  <tmNotice:goodsAndServices>
    Bardus populum circumdabit se cum captiosus populum.
    Smert populum circumdabit se cum captiosus populum qui eis differimus.
  </tmNotice:goodsAndServices>
  <tmNotice:notExactMatch>
    <tmNotice:udrp>
      <tmNotice:caseNo>D2003-0499</tmNotice:caseNo>
      <tmNotice:udrpProvider>WIPO</tmNotice:udrpProvider>
    </tmNotice:udrp>
```

</tmNotice:notExactMatch>

Lozano

Expires May 3, 2020

[Page 53]

```
</tmNotice:claim>
</tmNotice:notice>
```

For the formal syntax of the TCN please refer to [Section 7.1](#).

6.6. Sunrise List (SURL)

This section defines the format of the list containing every Domain Name Label (DNL) that matches a PRM eligible for Sunrise. The list is maintained by the TMDB and downloaded by Registries in regular intervals (see [Section 5.4.2.1](#)). The Registries use the Sunrise List during the Qualified Launch Program Period to check whether a requested DN matches a DNL of a PRM eligible for Sunrise.

The Sunrise List contains all the DNLs covered by a PRM eligible for Sunrise present in the TMDB at the datetime it is generated.

The Sunrise List is contained in a CSV formatted file that has the following structure:

- o first line: <version>,<Sunrise List creation datetime>

Where:

- + <version>, version of the file, this field MUST be 1.
- + <Sunrise List creation datetime>, date and time in UTC that the Sunrise List was created.

- o second line: a header line as specified in [[RFC4180](#)]

With the header names as follows:

DNL,insertion-datetime

- o One or more lines with: <DNL>,<DNL insertion datetime>

Where:

- + <DNL>, a Domain Name Label covered by a PRM eligible for Sunrise.
- + <DNL insertion datetime>, datetime in UTC that the DNL was first inserted into the Sunrise List. The possible two values of time for inserting a DNL to the Sunrise List are 00:00:00 and 12:00:00 UTC.

Example of a SURL

```
1,2012-08-16T00:00:00.0Z
DNL,insertion-datetime
example,2010-07-14T00:00:00.0Z
another-example,2012-08-16T00:00:00.0Z
anotherexample,2011-08-16T12:00:00.0Z
```

Figure 18

To provide authentication and integrity protection, the Sunrise List will be PGP signed by the TMDB (see also [Section 5.1.1.4](#)). The PGP signature of the Sunrise List can be found in the similar URI but with extension .sig as shown below.

The URL of the dy interface ([Section 4.3.3](#)) is:

- o < https://<tmdb-domain-name>/dnl/surl-latest.csv >
- o < https://<tmdb-domain-name>/dnl/surl-latest.sig >

7. Formal Syntax

7.1. Trademark Claims Notice (TCN)

The schema presented here is for a Trademark Claims Notice.

The BEGIN and END tags are not part of the schema; they are used to note the beginning and ending of the schema for URI registration purposes.

Copyright (c) 2016 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in [Section 4.c](#) of the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>).

BEGIN

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="urn:ietf:params:xml:ns:tmNotice-1.0"
  xmlns:tmNotice="urn:ietf:params:xml:ns:tmNotice-1.0"
  xmlns:mark="urn:ietf:params:xml:ns:mark-1.0"
```



```
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

<annotation>
  <documentation>
    Schema for representing a Trademark Claim Notice.
  </documentation>
</annotation>
<import namespace="urn:ietf:params:xml:ns:mark-1.0"/>
<element name="notice" type="tmNotice:noticeType"/>
<complexType name="holderType">
  <sequence>
    <element name="name" type="token" minOccurs="0"/>
    <element name="org" type="token" minOccurs="0"/>
    <element name="addr" type="tmNotice:addrType"/>
    <element name="voice" type="mark:e164Type" minOccurs="0"/>
    <element name="fax" type="mark:e164Type" minOccurs="0"/>
    <element name="email" type="mark:minTokenType" minOccurs="0"/>
  </sequence>
  <attribute name="entitlement" type="mark:entitlementType"/>
</complexType>
<complexType name="noticeType">
  <sequence>
    <element name="id" type="tmNotice:idType"/>
    <element name="notBefore" type="dateTime"/>
    <element name="notAfter" type="dateTime"/>
    <element name="label" type="mark:labelType"/>
    <element name="claim" type="tmNotice:claimType" minOccurs="0"
      maxOccurs="unbounded"/>
  </sequence>
</complexType>
<complexType name="claimType">
  <sequence>
    <element name="markName" type="token"/>
    <element name="holder" type="tmNotice:holderType"
      maxOccurs="unbounded"/>
    <element name="contact" type="tmNotice:contactType" minOccurs="0"
      maxOccurs="unbounded"/>
    <element name="jurDesc" type="tmNotice:jurDescType"/>
    <element name="classDesc" type="tmNotice:classDescType"
      minOccurs="0" maxOccurs="unbounded"/>
    <element name="goodsAndServices" type="token"/>
    <element name="notExactMatch" type="tmNotice:noExactMatchType"
      minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="jurDescType">
  <simpleContent>
    <extension base="token">
```



```
        <attribute name="jurCC" type="mark:ccType" use="required"/>
    </extension>
</simpleContent>
</complexType>
<complexType name="classDescType">
    <simpleContent>
        <extension base="token">
            <attribute name="classNum" type="integer" use="required"/>
        </extension>
    </simpleContent>
</complexType>
<complexType name="noExactMatchType">
    <choice maxOccurs="unbounded">
        <element name="udrp" type="tmNotice:udrpType"/>
        <element name="court" type="tmNotice:courtType"/>
    </choice>
</complexType>
<complexType name="udrpType">
    <sequence>
        <element name="caseNo" type="token"/>
        <element name="udrpProvider" type="token"/>
    </sequence>
</complexType>
<complexType name="courtType">
    <sequence>
        <element name="refNum" type="token"/>
        <element name="cc" type="mark:ccType"/>
        <element name="region" type="token" minOccurs="0"
            maxOccurs="unbounded"/>
        <element name="courtName" type="token"/>
    </sequence>
</complexType>
<complexType name="addrType">
    <sequence>
        <element name="street" type="token" minOccurs="1" maxOccurs="3"/>
        <element name="city" type="token"/>
        <element name="sp" type="token" minOccurs="0"/>
        <element name="pc" type="mark:pcType" minOccurs="0"/>
        <element name="cc" type="mark:ccType"/>
    </sequence>
</complexType>
<complexType name="contactType">
    <sequence>
        <element name="name" type="token"/>
        <element name="org" type="token" minOccurs="0"/>
        <element name="addr" type="tmNotice:addrType"/>
        <element name="voice" type="mark:e164Type"/>
        <element name="fax" type="mark:e164Type" minOccurs="0"/>
    </sequence>
</complexType>
```



```
<element name="email" type="mark:minTokenType"/>
</sequence>
<attribute name="type" type="mark:contactTypeType"/>
</complexType>
<simpleType name="idType">
  <restriction base="token">
    <pattern value="[a-fA-F0-9]{8}\d{1,19}"/>
  </restriction>
</simpleType>
</schema>
END
```

8. Acknowledgements

This specification is a collaborative effort from several participants in the ICANN community. Bernie Hoeneisen participated as co-author until version 02 providing invaluable support for this document. This specification is based on a model spearheaded by: Chris Wright, Jeff Neuman, Jeff Eckhaus and Will Shorter. The author would also like to thank the thoughtful feedback provided by many in the tmch-tech mailing list, but particularly the extensive help provided by James Gould, James Mitchell and Francisco Arias. This document includes feedback received from the following individuals: Paul Hoffman.

9. Change History

[[RFC Editor: Please remove this section.]]

9.1. Version 04

1. Ping update.

9.2. Version 05

1. Ping update.

10. IANA Considerations

This document uses URNs to describe XML namespaces and XML schemas conforming to a registry mechanism described in [[RFC3688](#)]. One URI assignment have been registered by the IANA.

Registration request for the Trademark Claims Notice namespace:

URI: urn:ietf:params:xml:ns:tmNotice-1.0

Registrant Contact: See the "Author's Address" section of this document.

XML: None. Namespace URIs do not represent an XML specification.

Registration request for the Trademark Claims Notice XML schema:

URI: urn:ietf:params:xml:ns:tmNotice-1.0

Registrant Contact: See the "Author's Address" section of this document.

XML: See [Section 7.1](#) of this document.

11. Security Considerations

This specification uses HTTP Basic Authentication to provide a simple application-layer authentication service. HTTPS is used in all interfaces in order to protect against most common attacks. In addition, the client identifier is tied to a set of IP addresses that are allowed to connect to the interfaces described in this document, providing an extra security measure.

The TMDB MUST provide credentials to the appropriate Registries and Registrars.

The TMDB MUST require the use of strong passwords by Registries and Registrars.

The TMDB, Registries and Registrars MUST use the best practices described in [RFC 7525](#) or its successors.

12. References

12.1. Normative References

[Claims50]

ICANN, "Implementation Notes: Trademark Claims Protection for Previously Abused Names", July 2013, <<https://newgtlds.icann.org/en/about/trademark-clearinghouse/previously-abused-16jul13-en.pdf>>.

[MatchingRules]

ICANN, "Memorandum on Implementing Matching Rules", September 2012, <<https://newgtlds.icann.org/en/about/trademark-clearinghouse/matching-rules-24sep12-en.pdf>>.

[QLP-Addendum]

ICANN, "Qualified Launch Program Addendum", April 2014, <<https://newgtlds.icann.org/en/about/trademark-clearinghouse/rpm-requirements-qlp-addendum-10apr14-en.pdf>>.

[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>>.

[RFC3688] Mealling, M., "The IETF XML Registry", [BCP 81](#), [RFC 3688](#), DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.

[RFC7848] Lozano, G., "Mark and Signed Mark Objects Mapping", [RFC 7848](#), DOI 10.17487/RFC7848, June 2016, <<https://www.rfc-editor.org/info/rfc7848>>.

[RPM-Requirements]

ICANN, "Rights Protection Mechanism Requirements", September 2013, <<https://newgtlds.icann.org/en/about/trademark-clearinghouse/rpm-requirements-30sep13-en.pdf>>.

12.2. Informative References**[ICANN-GTLD-AGB-20120604]**

ICANN, "gTLD Applicant Guidebook Version 2012-06-04", June 2012, <<http://newgtlds.icann.org/en/applicants/agb/guidebook-full-04jun12-en.pdf>>.

[ISO3166-2]

ISO, "International Standard for country codes and codes for their subdivisions", 2006, <http://www.iso.org/iso/home/standards/country_codes.htm>.

[RFC2818] Rescorla, E., "HTTP Over TLS", [RFC 2818](#), DOI 10.17487/RFC2818, May 2000, <<https://www.rfc-editor.org/info/rfc2818>>.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", [RFC 3339](#), DOI 10.17487/RFC3339, July 2002, <<https://www.rfc-editor.org/info/rfc3339>>.

[RFC4180] Shafranovich, Y., "Common Format and MIME Type for Comma-Separated Values (CSV) Files", [RFC 4180](#), DOI 10.17487/RFC4180, October 2005, <<https://www.rfc-editor.org/info/rfc4180>>.

- [RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", [RFC 4648](#), DOI 10.17487/RFC4648, October 2006, <<https://www.rfc-editor.org/info/rfc4648>>.
- [RFC4880] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", [RFC 4880](#), DOI 10.17487/RFC4880, November 2007, <<https://www.rfc-editor.org/info/rfc4880>>.
- [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, <<https://www.rfc-editor.org/info/rfc5280>>.
- [RFC5890] 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/info/rfc5890>>.
- [RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", [RFC 7230](#), DOI 10.17487/RFC7230, June 2014, <<https://www.rfc-editor.org/info/rfc7230>>.
- [RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", [RFC 7231](#), DOI 10.17487/RFC7231, June 2014, <<https://www.rfc-editor.org/info/rfc7231>>.
- [RFC7235] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Authentication", [RFC 7235](#), DOI 10.17487/RFC7235, June 2014, <<https://www.rfc-editor.org/info/rfc7235>>.
- [RFC7719] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", [RFC 7719](#), DOI 10.17487/RFC7719, December 2015, <<https://www.rfc-editor.org/info/rfc7719>>.
- [WIPO-NICE-CLASSES] WIPO, "WIPO Nice Classification", 2015, <<http://www.wipo.int/classifications/nice/en>>.
- [WIPO.ST3] WIPO, "Recommended standard on two-letter codes for the representation of states, other entities and intergovernmental organizations", March 2007, <http://www.iso.org/iso/home/standards/country_codes.htm>.

Author's Address

Gustavo Lozano
ICANN
12025 Waterfront Drive, Suite 300
Los Angeles 90292
US

Phone: +1.3103015800

Email: gustavo.lozano@icann.org