IT Department
Documentation

Pre-Delegation Testing

IDN Test Cases

Version PA6

File name: PDT_IDN_TC.docx
Last saved: 2013-03-04

The Internet Infrastructure Foundation
PO Box 7399, SE-103 91 Stockholm Tel +46(0)8-452 35 00 Fax +46(0)8-452 35 02 VAT no SE802405019001 www.iis.se

Document control

Document information and security

Made by	Responsible for fact	Responsible for document
Cary Karp	Cary Karp	Cary Karp

Security class	File name
External	PDT_IDN_TC.docx

Revisions

Date	Version	Name	Description
2013-01-13	PA1	Cary Karp	Initial document
2013-02-07	PA2	Cary Karp	Input requirements clarified; Several EPP tests segregated into new TC
2013-02-08	PA3	Rickard Bellgrim	Update text and rearrange test cases
2013-02-11	PA4	Lennart Bonnevier	Review text
2013-03-04	PA5	Cary Karp	Modified in response to editorial review
2012-03-04	PA6	Rickard Bellgrim	Verify input data against the application

LIST OF CONTENTS

1. Introduction

1.1 Scope

The Pre-Delegation Testing Provider will verify that each IDN table is formatted according to RFC 4290 or RFC 3743 or a local format for which ICANN has given dispensation; that each listed code point is valid under the IDNA protocol; that every string of tabulated code points that can be registered as a label conforms to the IDN Guidelines: that all policy and context-dependent requirements of IDNA and the Guidelines are clearly stated and enforced in the registry; and that all policies associated with the management of variant relationships between tabulated code points are similarly documented and enforced.

1.2 References

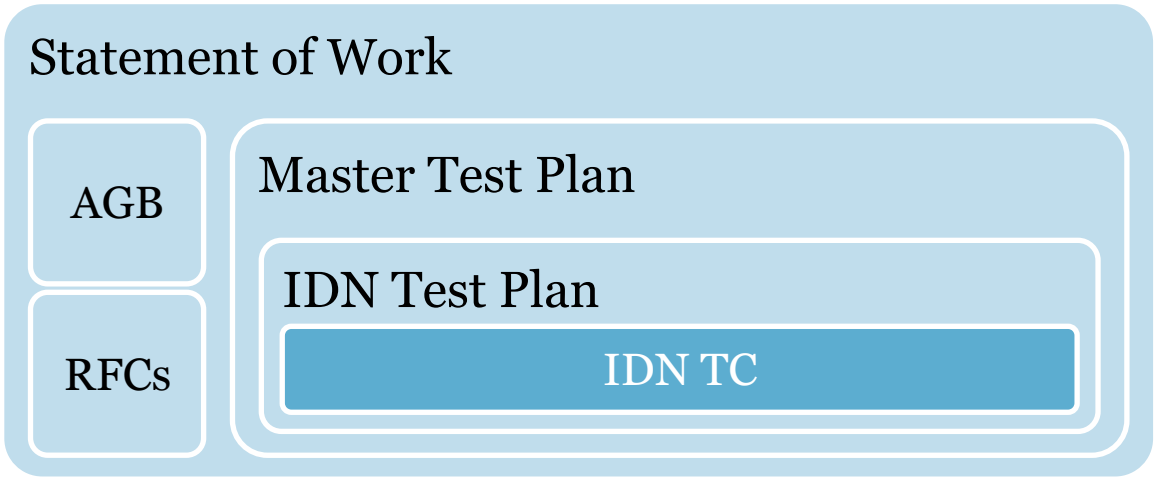
1.2.1 External

- ICANN gTLD Applicant Guidebook, Version 2012-06-04
- ICANN Guidelines for the Implementation of Internationalized Domain Names, Version 3.0 (“IDN Guidelines”)
- IEEE 829-2008
- RFC 3743
- RFC 4290
- RFCs 5890 through 5894 (“IDNA”)
- The Unicode Standard (“Unicode”)

1.2.2 Internal

- Pre-Delegation Testing, Statement of Work
- Pre-Delegation Testing, Master Test Plan
- Pre-Delegation Testing, IDN Test Plan

1.2.3 Document Hierarchy



1.3 Context

The tests have two basic elements. The first is the offline review of a table and the associated policy statements. The second is determining the registry response to the attempted registration of labels constructed specifically to verify that code points and strings not permitted for registration are rejected, and those that are permitted are accepted, with particular attention to contextual constraints imposed in the reference documents and the additional normative instruments they invoke. Where expedient, the online test components are aggregated into a single test, IDNvalid08.

All tests will utilize algorithmic support to the extent possible. However, anticipated variation in the tabulation of nominally identical script repertoires and the substance and format of the associated policy statements, will likely necessitate a significant amount of manual testing.

The tests call for the generation of a number of test labels. This may be part of the action of the test operators but in many instances, suitable labels will be have been pre-generated by the IDN Test Officer. If so, the only decision necessary will be which of the available test labels to use. In cases where a test label needs to be custom designed, it will be added to the repository for reuse where appropriate in subsequent testing. Although there is some preliminary expectation on the basis of public portions of the applications, the test label repository cannot be seeded definitively until the first actual test objects have been forwarded by ICANN.

1.4 Notation for description

Each IDN test case is described under a separate heading, below. The test procedures are described with the test case to which they apply.

2. IDNvalid01

2.1 Test case identifier

IDNvalid01 - IDN table validation

2.2 Objective

This test verifies that the format of a code point table conforms to RFC 4920 or RFC 3743.

2.3 Inputs

The following information will be needed as input for this test case:

Id	Description	Type
TestTable	The table under scrutiny.	File
LocalTableFormat	Describes the table format if it does not comply with either of the reference RFCs.	File
LocalTableJustification	Verifiable warrant for using a local format instead of either of the reference RFCs.	File

Neither of the reference RFCs specifies a rigorous enough format for TestTable to be automatically parsed for conformance, and there is no way to predict the details of an instance of LocalTableFormat. Manual examination of a table is necessary, in any case, to determine whether relevant policy statements are interposed with the structured code point listings as narrative commentary. The validation of format compliance can easily be conducted during the course of that examination, leaving no reason for segregating the ordered component of the table for subsequent automated processing.

2.4 Outcome(s)

The response to this test will be a pass/fail determination.

2.5 Environmental needs

- Basic desktop

2.6 Special procedural requirements

The person running this test must understand the elements of an IDN table format, both as described in the reference RFCs, and in order to assess the sufficiency of a locally defined alternative and the justification for its use.

2.7 Intercase dependencies

None.

2.8 Ordered description of steps to be taken to execute the test case

1. Verify the all IDN scripts/languages in the application are provided as input data to this test.
2. Compare the TestTable format with the one prescribed in RFC 4920 and if it conforms end the test as passed. If it does not conform proceed to the next step.
3. Compare the TestTable format with the one prescribed in RFC 3743 and if it conforms end the test as passed. If it does not conform proceed to the next step.
4. Assess the sufficiency of LocalTableJustification. If it adequately explains why neither of the reference RFC formats can be used, and LocalTableFormat supports the functionality necessary to conduct the other tests, end the test as passed. If these criteria are not met, end the test as failed.

3. IDNvalid02

3.1 Test case identifier

IDNvalid02 - IDNA code point validation

3.2 Objective

This test verifies that the status of each tabulated code point is PROTOCOL VALID (PVALID) or CONTEXTUAL RULE REQUIRED (CONTEXTn) as defined in RFC 5892 when its algorithms are applied to the Unicode Standard, version 6.2.

3.3 Inputs

The following information will be needed as input for this test case:

Id	Description	Type
TestTable	See Section 2.3, above.	File
AvailableCodepointTable	A tabular listing of all PVALID and CONTEXTn code points in Unicode 6.2, with separate columns indicating the IDN status and the Unicode script property value for the code point that keys every row. This file is provided internally.	File

3.4 Outcome(s)

The output will be an extended version of TestTable with new columns added for IDN status and Unicode script property values. This will be assigned the ID ExtendedTestTable and used as input for subsequent tests. There will also be a pass/fail determination.

3.5 Environmental needs

- Text sorting and comparison utilities

3.6 Special procedural requirements

None.

3.7 Intercase dependencies

IDNvalid01

3.8 Ordered description of steps to be taken to execute the test case

For every row in TestTable, keyed on the first code point appearing in it, determine if there is a corresponding row in AvailableCodepointTable, and if there is, generate ExtendedTestTable and end the test as passed. If any row is keyed with a code point that does not also key a row in AvailableCodepointTable, end the test as failed.

4. IDNvalid03

4.1 Test case identifier

IDNvalid03 - IDNA Context Rule Validation

4.2 Objective

This test verifies that a tabulated code point with the status CONTEXTJ or CONTEXTO can only be used according to the contextual rule for that code point given in RFC 5892.

4.3 Inputs

The following information will be needed as input for this test case:

Id	Description	Type
ExtendedTestTable	Table generated by test IDNvalid02.	File
IDNvalid08Results	Verification of online registry processing.	Boolean

IDNAContextualRules is the contextual rules listed in RFC 5892, Appendix A.

4.4 Outcome(s)

The response to this test will be a pass/fail determination.

4.5 Environmental needs

- Basic desktop

4.6 Special procedural requirements

The person conducting this test must understand the application of the CONTEXTn rules in the reference RFC.

4.7 Intercase dependencies

This test is dependent on output from IDNvalid08.

4.8 Ordered description of steps to be taken to execute the test case

1. If the IDN property CONTEXTO or CONTEXTJ does not appear in any row in ExtendedTestTable this test is inapplicable and can be terminated with the test as passed. If either of these properties appears, proceed to the next step.
2. Obtain the results for the IDNvalid08 tests of labels TL1, TL2, and TL3.
3. If TL1 was rejected end the test as failed.
4. If TL2 was accepted end the test as failed.
5. If TL3 was accepted, end the test as failed.
6. If TL3 was rejected, end the test as passed.

5. IDNvalid04

5.1 Test case identifier

IDNvalid04 - IDN script validation.

5.2 Objective

This test verifies that the code point array in a single table is restricted to a single script with an explicit script property value as defined in the Unicode Standard Annex #24, and that code points with COMMON and INHERITED script property values are correctly associated with the designated script.

5.3 Inputs

The following information will be needed as input for this test case:

Id	Description	Type
ExtendedTestTable	Table generated by test IDNvalid02.	File
AvailableCodepointTable	A tabular listing of all PVALID and CONTEXTn code points in Unicode 6.2, with separate columns indicating the IDN status and the Unicode script property value for the code point that keys every row. This file is provided internally.	File
ScriptIntegrityPolicies	The script integrity policies declared by the registry.	File
IDNvalid08Results	Verification of online registry processing.	Boolean

UAX24 is the Unicode Standard Annex #24; Unicode Script Property.

5.4 Outcome(s)

The response to this test will be a pass/fail/warning determination.

5.5 Environmental needs

- Basic desktop

5.6 Special procedural requirements

The person conducting this test must understand Unicode script properties designating specific scripts, as well as the values COMMON and INHERITED. These are described in Unicode Standard Annex #24 (UAX #24) which states that COMMON and INHERITED are assigned to code points that are used with more than one script, but that this does not imply usability with all scripts. UAX #24 does not provide unequivocal guidance on how such restrictions are to be applied but does include a number of test cases illustrating correct and incorrect use of those properties. These tests will be applied to all IDN tables that include COMMON and INHERITED, with the exception noted in the following paragraph, and the result will serve as a pass/fail criterion.

The only code points with the COMMON script property that will be accepted in any IDN table are 0030..0039 DIGIT ZERO..DIGIT NINE, and U+002D HYPEN-MINUS. This is the digit and hyphen component of the basic ASCII LDH repertoire and will be referred to as "DH" in the following text.

A test table that indiscriminately accepts all code points with COMMON and INHERITED script properties for use with all scripts handles those properties incorrectly and will result in the termination of the test with a warning. If the proper implementation of these script property values had been an explicit criterion in the DoW, the blanket support for COMMON and INHERITED would result in failure.

The IDN Test Officer will collate the UAX #24 test cases in a format that can readily be applied to the present test and the failure of any of them will result in the entire test failing, rather than a warning.

5.7 Intercase dependencies

This test is dependent on output from IDNvalid08.

5.8 Ordered description of steps to be taken to execute the test case

1. If the column in ExtendedTestTable indicating the script property value contains the same explicit script designator for every row in the table, but the table is not correctly labeled as supporting that script, issue a warning and proceed to the next step. If the table is correctly labeled, proceed to the next step.
2. Obtain the results for the IDNvalid08 tests of labels TL4, TL5, and TL6, and the UAX24 tests.
3. If TL4 was rejected end the test as failed.
4. If TL5 was accepted end the test as failed.
5. If TL6 was accepted end the test as failed.
6. If TL6 was rejected end the test as passed.
7. If the only other script property value appearing in the table is COMMON, and every code point it is assigned to is in the DH cluster, end the test as passed.
8. If the value COMMON (assigned to non-DH code points) or INHERITED appears, and any of the UAX24 tests have failed, end the test as failed.
9. If every row in AvailableCodepointTable with the COMMON and INHERITED script property value corresponds to a row in ExtendedTestTable, terminate the test with a warning.

6. IDNvalid05

6.1 Test case identifier

IDNvalid05 - IDN script-mixing rule validation

6.2 Objective

This test verifies that a table including code points from more than one script is associated with rules that either prevent the mixing of those scripts in individual labels, or restrict such mixing as specified in the IDN Guidelines. It also verifies that each script is in a clearly partitioned block that is effectively equivalent to a separate table meeting the criteria set in IDNvalid04.

6.3 Inputs

The following information will be needed as input for this test case:

Id	Description	Type
ExtendedTestTable	Table generated by test IDNvalid02.	File
ScriptIntegrityPolicies	The script integrity policies declared by the registry.	File
IDNvalid08Results	Verification of online registry processing.	Boolean

6.4 Outcome(s)

The response to this test will be a pass/fail/warning determination.

6.5 Environmental needs

- Basic desktop

6.6 Special procedural requirements

None.

6.7 Intercase dependencies

This test provides input to, and is dependent on output from, IDNvalid08.

6.8 Ordered description of steps to be taken to execute the test case

1. If the column in ExtendedTestTable indicating script property value contains more than one explicit script designator, determine that the table is subdivided into clearly labeled blocks and that each is restricted to code points with the indicated explicit script property value. If it is not, end the test as failed. If it is correctly subdivided into blocks proceed to the next step.
2. Treat each block as a discrete table and pass them to the operator of IDNvalid08. Perform the IDNvalid04 steps on the returned data for each of the submitted tables. If any failure conditions are returned, terminate the test accordingly.
3. Determine if ScriptIntegrityPolicies either explicitly prohibits the intermingling of elements from each of the blocks in a single label, or explains and justifies the conditions

under which such intermingling is permitted. If there is no such statement, end the test as failed. If there is such a statement, end the test as passed.

7. IDNvalid06

7.1 Test case identifier

IDNvalid06 - IDN language validation

7.2 Objective

This test verifies that a table associated with a language rather than a script is consistent with the script-based constraints in the preceding test cases, and that linguistic warrant is demonstrated in any policy statement permitting the intermingled use of multiple scripts in individual labels.

7.3 Inputs

The following information will be needed as input for this test case:

Id	Description	Type
ExtendedTestTable	Table generated by test IDNvalid02.	File
LanguagePolicies	The language support policies declared by the registry.	File

7.4 Outcome(s)

The response to this test will be a pass/fail determination.

7.5 Environmental needs

- Basic desktop

7.6 Special procedural requirements

The person conducting this test must be familiar with basic concepts of writing systems.

7.7 Intercase dependencies

None.

7.8 Ordered description of steps to be taken to execute the test case

If the column in ExtendedTestTable indicating script property value contains more than one explicit script designator and the table is declared to support a language with a writing system that uses all of those scripts, determine that LanguagePolicies provides verifiable linguistic warrant for that assertion. If it does not, terminate the test as failed. If there is such warrant, terminate the test as passed.

NOTE: the only writing system thus far figuring in the discussion of IDN repertoires that uses multiple scripts is Japanese, which intermingles elements of the Han, Hiragana, Katakana, and the Basic Latin scripts.

8. IDNvalid07

8.1 Test case identifier

IDNvalid07 - IDN variant code point validation

8.2 Objective

This test verifies that the rules given for the processing of variant relationships between listed code points are enforced.

8.3 Inputs

The following information will be needed as input for this test case:

Id	Description	Type
ExtendedTestTable	Table generated by test IDNvalid02.	File
VariantAlgorithms	The variant generation algorithms used by the registry.	File
VariantPolicies	The variant management policies declared by the registry.	File

8.4 Outcome(s)

The response to this test will be a pass/fail/warning determination.

8.5 Environmental needs

- EPP client
- IPv4 or IPv6 connectivity.

8.6 Special procedural requirements

The person conducting this test must understand the concept of variant code points that ICANN applies to IDN repertoires and the associated registration policies. Further procedural constraints are discussed in Section 2.5.2 of the IDN Test Plan.

8.7 Intercase dependencies

None.

8.8 Ordered description of steps to be taken to execute the test case

1. If no row in ExtendedTestTable includes more than one code point, and no row indicates any correlation between its code point and a code point on any other row, this test is inapplicable and can be terminated.
2. If correlations between two code points are indicated ("variant relationships"), verify either that VariantPolicies explains each such relationship and the constraints that attach to it, or that VariantAlgorithms describes the processing of each such relationship. If neither is available, end the test as failed. If these statements exist but do not adequately indicate the behavior that can be expected at the registry, issue a warning and proceed to the next step.
3. Construct a test label including a code point that is expected to be replaced by another code point in the registry and submit a request for its registration. If the request is accepted without any indication of that transformation having been applied, end the test as failed. If the correct transformation has been applied proceed to the next step.
4. Perform the same test iteratively for at least two further code points for which variant substitution is expected. If any of these sub-tests fail, end the test as failed. If they all pass proceed to the next step.
5. Determine if any component of the variant management process blocks the registration of a label in one variant form if a label in another variant form has already been registered. Construct a test label in a form that should cause such blocking and submit it for registration. Then construct a second test label in the form that is expected to be blocked and submit it for registration. If the second label is accepted end the test as failed. If it is rejected, repeat this test at least twice in the same iterative manner as with the preceding step. If the registry accepts and rejects labels in accordance with the anticipated behavior on each iteration, end the test as passed. If the expected behavior is not observed, end the test as failed.

9. IDNvalid08

9.1 Test case identifier

IDNvalid08 - IDN online registry response verification

9.2 Objective

This test verifies that test strings needed for preceding tests are correctly processed by the online registry.

9.3 Inputs

The following information will be needed as input for this test case:

Id	Description	Type
ExtendedTestTable	Table generated by test IDNvalid02.	File
IDNvalid05SubTables	Segmented test tables submitted by the IDNvalid05 operator.	File
ScriptIntegrityPolicies	The script integrity policies declared by the registry.	File

9.4 Outcome(s)

The response to this test will be a pass/fail/warning determination.

9.5 Environmental needs

- EPP client
- IPv4 or IPv6 connectivity

9.6 Special procedural requirements

None.

9.7 Intercase dependencies

This test receives input from IDNvalid05, and sends output to IDNvalid03, IDNvalid04, IDNvalid05.

9.8 Ordered description of steps to be taken to execute the test case

1. For every row in ExtendedTestTable that lists the IDN property CONTEXTO or CONTEXTJ, construct three test labels including the code point with that property. The first label, TL1 will place the code point in the context required by the associated rule. The second, TL2, will place the code point in a context that violates the rule. The third, TL3, will include two instances of the code point, of which one will respect the contextual rule and the other will violate it.
2. Submit a request to register TL1. If it is rejected, note this in the test report.
3. Submit a request to register TL2. If it is accepted, note this in the test report.
4. Submit a request to register TL3. If it is accepted, note this in the test report.
5. Construct a test label consisting solely of listed code points, TL4, and submit a request to register it. If it is rejected, note this in the test report.
6. Construct a test label that includes at least one code point that is not listed, TL5, and submit a request to register it. If it is accepted, note this in the test report.
7. Construct a test label that includes at least one code point with an explicit script property value that both differs from any listed in the table, and is not allowed in ScriptIntegrityPolicies, TL6. Submit a request to register it. If it is accepted, note this in the test report.
8. If the script property value COMMON is assigned to non-DH code points or the script property value INHERITED appears, run the UAX24 tests and if any of them fail, note that the UAX24 tests have failed.
9. Process each table included in IDNvalid05Subtables as a separate ExtendedTestTable, iteratively performing Steps 1-8 on each, incrementing the TL counter accordingly.

10. Global

10.1 Glossary

The glossary is available in the Master Test Plan.

10.2 Document change procedures

Document change procedures are documented in the Master Test Plan.