Internationalized Domain Names support in POSIX getaddrinfo
draft-josefsson-getaddrinfo-idn-00

Abstract

This document describes an extension for Internationalized Domain
Names support in the POSIX getaddrinfo function.

Status of this Memo

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

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

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

This Internet-Draft will expire on December 17, 2010.

Copyright Notice

Copyright (c) 2010 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
(http://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. Preface ........................................... 3
2. Background ....................................... 3
3. Alternative Approaches ............................ 3
4. What I propose ..................................... 4
5. Details ........................................... 4
6. Status ........................................... 5
7. Future ........................................... 5
8. Feedback ......................................... 5
9. Security Considerations ........................... 5
10. IANA Considerations ............................... 5
11. Acknowledgements ................................... 5
12. Normative References ............................... 5
Author's Address ..................................... 5
1. Preface

This document was originally written in 2003 and published and implemented as part of GNU Libidn. This is a copy of the memo but in IETF form. The document was written informally, and does not (yet?) follow typical IETF document formats. The intention is to make the IETF community aware of this work, to see if there are interest in the ideas.

2. Background

Libidn is a package for internationalized string handling based on the Stringprep, Punycode and Internationalized Domain Names in Applications (IDNA) specifications. It can be used by applications directly by linking to it, as is done by, e.g., Gnus, KDE, and Mutt.

Having each and every application link with and perform its own IDN handling is not a good idea. It bloats the code and makes things unnecessarily complex. Only few applications, such as web browsers and mail clients, will need to do this in the future, to provide good user interfaces for internationalization.


3. Alternative Approaches

There are implementation that modify gethostbyname() to accept UTF-8 strings and perform the IDNA ToASCII operation within gethostbyname().

There are even implementations that assume gethostbyname (on the client host) perform no validation of the string and will send UTF-8 strings out to the DNS server, and perform the IDN-conversion on the DNS server.
Some doubts can be raised whether this is an approach that is likely to be standardized. It also lack in functionality: it only provide black-box ToASCII functionality. The application cannot extract the output from the ToASCII operation. More important, there is no way to perform a ToUnicode operation that applications may want to use for display purposes. Furthermore, while the first can support locale specific character sets (e.g., ISO-8859-1), the second approach is bound to either guess the character set, or always use UTF-8.

See also the thread rooted in <iluel7n6bmufsf@latte.josefsson.org>

4. What I propose

The getaddrinfo() API should have two new flags, AI_IDN and AI_CANONIDN. Roughly they correspond to IDNA ToASCII and IDNA ToUnicode, but there are several details. Note that strings are still 'char*', i.e. it does not use the "wide" character type, and that the encoding of non-ASCII strings are the current locale's character set (i.e., nl_langinfo(CODESET)).

An application that uses AI_IDN signal to the getaddrinfo() implementation that the input host name may be non-ASCII and that the appropriate IDNA ToASCII steps should be carried out on the input, and the output from the ToASCII operation (if any) should be used in the lookup using the current resolver processing.

An application that uses AI_CANONIDN signal to the getaddrinfo() implementation that the input host name should be put through the IDNA ToUnicode steps, and the output of that placed in the 'ai_canonname' field of the resulting structure. Normal resolver processing applies to the input string, of course.

Consequently, an application that uses AI_IDN|AI_CANONIDN signal to the getaddrinfo() implementation that the input host name may be non-ASCII and should be put through the IDNA ToASCII steps before run through the resolver, and that the input string should also be run through the IDNA ToUnicode steps and the output of that placed in the
'ai_canonname' field.

The semantics of AI_CANONNAME|AI_CANONIDN is that instead of running the ToUnicode IDNA steps on the input string, the canonical host name as returned by the resolver for the input string should be used in the ToUnicode IDNA step.

5. Details

Four new flags has been proposed; AI_IDN_ALLOW_UNASSIGNED, AI_IDN_USE_STD3_ASCII_RULES for getaddrinfo, and NI_IDN_ALLOW_UNASSIGNED, NI_IDN_USE_STD3_ASCII_RULES for getnameinfo. The implementation is simple, if specified those flag will set the appropriate flag in the call to the IDNA functions. See the RFC for the meaning of those flags.

6. Status

The AI_IDN flag has been implemented and shipped as a proof-of-concept patch for GNU Libc with GNU Libidn since January 2003. Binary libc packages with the patch exists for (at least) two GNU/Linux distributions. The AI_CANONIDN flag is not yet implemented.

As of March 2004, Libidn has been integrated as an add-on in the GNU Libc CVS repository. The AI_CANONIDN flag has been implemented. The AllowUnassigned and UseSTD3ASCIIRules flags were added.

7. Future

Allow non-ASCII in gethostname (and similar functions), if administrator has supplied, e.g., 'option idn' in /etc/resolv.conf?

8. Feedback

This document is a work-in-progress and the details may change. Contact me at simon@josefsson.org to discuss changes.
9. Security Considerations

TBA.

10. IANA Considerations

TBA.

11. Acknowledgements

Ulrich Drepper integrated the work in GNU Libc.

12. Normative References