INTERNET-DRAFT
<draft-ietf-krb-wg-info-ascii-gen-string-00.txt>
November 13, 2001
Expires: May 13, 2002

Jeffrey Altman
Columbia University

Informational: Kerberos GeneralString to be Interpreted as ASCII Only

Status of this Memo

Abstract:

To ensure future interoperability between existing deployments of Kerberos 5 (RFC 1510) and future standards efforts the Kerberos Working Group strongly recommends that users of Kerberos 5 implementations SHOULD NOT deploy Kerberos principal or service names that utilize characters not included in the 94 printable characters specified in the International Reference Version of ISO-646/ECMA-6 (aka U.S. ASCII).

Background:

The original specification of the Kerberos protocol in RFC 1510 uses GeneralString in numerous places for human-readable string data. Historical implementations of Kerberos cannot utilize the full power of GeneralString. This ASN.1 type requires the use of designation and invocation escape sequences as specified in ISO-2022/ECMA-35 to switch character sets, and the default character set that is designated for G0 is the ISO-646/ECMA-6 International Reference Version (IRV) (aka U.S. ASCII), which mostly works.

ISO-2022/ECMA-35 defines four character-set code elements (G0..G3) and two Control-function code elements (C0..C1). DER prohibits the invocation of character sets into any but the G0 and C0 sets. Unfortunately, this seems to have the side effect of prohibiting the use of ISO-8859 (ISO Latin) character-sets or any other character-sets that utilize a 96-character set, since it is prohibited by ISO-2022/ ECMA-35 to invoke them into the G0 code element.

In practice, many implementations treat GeneralStrings as if they were 8-bit strings of whichever character set the implementation defaults to, without regard for correct usage of character-set designation escape sequences. The default character set is often determined by the current user's operating system dependent locale. At least one major implementation places unescaped UTF-8 encoded Unicode characters in the GeneralString. This failure to adhere to the GeneralString specifications results in interoperability issues when conflicting character encodings are utilized by the Kerberos clients, services, and KDC.

This unfortunate situation is the result of improper documentation of the restrictions of the ASN.1 GeneralString type in prior Kerberos specifications.

Transitioning to the use of UTF-8:

For various reasons, a transition to the use of UTF-8 encoding is desirable. First, there is a mandate from the IESG to support international character sets generally, and UTF-8 specifically. Also, the fact that there are existing installations violating the ISO-646/ECMA-6 restrictions and accepting the resulting pain indicates that there is a clear need to support alternate character sets in princpal names and passwords. As I8N support is deployed in DNS there will be a need to represent Unicode service names.

At the same time, backward compatibility with the existing installed base is crucial. Few site administrators have the luxury of declaring a flash cut-over of all users, applications, servers, etc to an incompatible protocol -- many have non-local users over whom they have little or no control. To this end, it is important for new implementations to be able to tell whether a particular non-US-ASCII string was encoded as UTF-8 by a new implementation, or as something else by an old implementation. In the latter case, it is of course impossible to know what the "something else" is without being told in advance.

There have been three proposals for how the fields currently encoded as GeneralStrings should be interpreted in order to accomplish such a transition:

(1) Lie. Start using UTF-8, but continue to encode all of these fields as GeneralStrings. To my knowledge, this is what Microsoft is doing today. This approach is attractive because it requires no changes to the message format specification and provides 100% compatibility with deployments that adhere to the ISO-646/ECMA-6 standards. However, it has several key problems. First, it does not allow a new implementation to tell whether a particular string was encoded as UTF-8 by a post-RFC-1510 implementation or as some 8-bit local character set by an older implementation. Second, there are potential problems with encoding arbitrary 8-bit strings, particularly for those who are using off-the-shelf ASN.1 compilers. Finally, violating the ASN.1 specification in this manner would be unpopular with the ITU which is a serious issue.

(2) Don't lie. Start using UTF-8 encoded in GeneralStrings with ISO-2022/ECMA-35 compatible escape sequences. While this has the appearance of following the ASN.1 specification for GeneralString, it has the problem that UTF-8 cannot be legally encoded due to the restriction that only G0 compatible character-set can be specified. This creates problems for implementors using off-the-shelf ASN.1 compilers as well as political issues with the ITU.

(3) Don't use GeneralString. In all the places where we currently use GeneralString, begin using a new "KerberosString" type instead. This type would be defined as an ASN.1 choice, with GeneralString and some form of UTF-8 strings as alternatives. The selection of which alternative to use would be based on whether one was talking to an old implementation or a new one. This approach does involve changing the message format _specifications_, but as long as the GeneralString choice is used, the actual ASN.1 DER encoding does not change. There is a transition issue in that replacing a type with a choice containing that type is not always a legitimate thing to do, but as long as DER are used (which is always the case in Kerberos 5), it does work correctly.

    The new KerberosString could be implemented as one of:

```
KerberosString ::= CHOICE {
                     general GeneralString (VisibleString),
                     utf8 UTF8String
}
```

    or as

```
KerberosString ::= CHOICE {
                     general GeneralString (VisibleString),
                     ...
}
```

In both cases, most (if not all) occurrences of GeneralString would be replaced with the new KerberosString.

It is the belief of the Kerberos Working group that regardless of the final decision that is reached on how to transition to the use of UTF-8 those implementors and deployments which have restricted their use of character-sets to the ISO-646/ECMA-6 IRV will have significantly fewer difficulties making the transition. This is because the IRV is a proper subset of the UTF-8 encoding.

Security Considerations:

Interoperability conflicts can result in denial of service for clients that utilize character-sets in Kerberos strings other than those stored in the KDC database.

References:

RFC-1510	The Kerberos Network Authentication Service (V5)
ISO-646/ECMA-6	7-bit Coded Character Set
ISO-1022/ECMA-35	Character Code Structure and Extension Techniques
ISO-4873/ECMA-43	8-bit Coded Character Set Structure and Rules
RFC-2279	UTF-8, a transformation format of ISO-10646

Acknowledgements:

This document while edited by Jeffrey Altman <jaltman@columbia.edu> (Columbia University) was directly derived from e-mail discussions with Jeffrey T. Hutzelman <jhutz+@cmu.edu> (CMU) and Tom Yu <tlyu@mit.edu> (MIT).