                    IMAP4 Extension for Fuzzy Search

Abstract

   This document describes an IMAP protocol extension enabling a server
   to perform searches with inexact matching and assigning relevancy
   scores for matched messages.

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Further information on
   Internet Standards is available in Section 2 of RFC 5741.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc6203.

1. Introduction

   When humans perform searches in IMAP clients, they typically want to
   see the most relevant search results first. IMAP servers are able to
   do this in the most efficient way when they're free to internally
   decide how searches should match messages. This document describes a
   new SEARCH=FUZZY extension that provides such functionality.

2. Conventions Used in This Document

   In examples, "C:" indicates lines sent by a client that is connected
   to a server. "S:" indicates lines sent by the server to the client.

   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 RFC 2119 [KEYWORDS].

3. The FUZZY Search Key

   The FUZZY search key takes another search key as its argument. The
   server is allowed to perform all matching in an implementation-
   defined manner for this search key, including ignoring the active
   comparator as defined by [RFC5255]. Typically, this would be used to
   search for strings. For example:

      C: A1 SEARCH FUZZY (SUBJECT "IMAP break")
      S: * SEARCH 1 5 10
      S: A1 OK Search completed.

   Besides matching messages with a subject of "IMAP break", the above
   search may also match messages with subjects "broken IMAP", "IMAP is
   broken", or anything else the server decides that might be a good
   match.

   This example does a fuzzy SUBJECT search, but a non-fuzzy FROM
   search:

      C: A2 SEARCH FUZZY SUBJECT work FROM user@example.com
      S: * SEARCH 1 4
      S: A2 OK Search completed.

   How the server handles multiple separate FUZZY search keys is
   implementation-defined.

   Fuzzy search algorithms might change, or the results of the
   algorithms might be different from search to search, so that fuzzy
   searches with the same parameters might give different results for
   1) the same user at different times, 2) different users (searches

executed simultaneously), or 3) different users (searches executed at
different times). For example, a fuzzy search might adapt to a
user's search habits in an attempt to give more relevant results (in
a "learning" manner). Such differences can also occur because of
operational decisions, such as load balancing. Clients asking for
"fuzzy" really are requesting search results in a not-necessarily-
deterministic way and need to give the user appropriate warning about
that.

4. Relevancy Scores for Search Results

   Servers SHOULD assign a search relevancy score for each matched
   message when the FUZZY search key is given. Relevancy scores are
   given in the range 1-100, where 100 is the highest relevancy. The
   relevancy scores SHOULD use the full 1-100 range, so that clients can
   show them to users in a meaningful way, e.g., as a percentage value.

   As the name already indicates, relevancy scores specify how relevant
   to the search the matched message is. It's not necessarily the same
   as how precisely the message matched. For example, a message whose
   subject fuzzily matches the search string might get a higher
   relevancy score than a message whose body had the exact string in the
   middle of a sentence. When multiple search keys are matched fuzzily,
   how the relevancy score is calculated is server-dependent.

   If the server also advertises the ESEARCH capability as defined by
   [ESEARCH], the relevancy scores can be retrieved using the new
   RELEVANCY return option for SEARCH:

      C: B1 SEARCH RETURN (RELEVANCY ALL) FUZZY TEXT "Helo"
      S: * ESEARCH (TAG "B1") ALL 1,5,10 RELEVANCY (4 99 42)
      S: B1 OK Search completed.

   In the example above, the server would treat "hello", "help", and
   other similar strings as fuzzily matching the misspelled "Helo".

   The RELEVANCY return option MUST NOT be used unless a FUZZY search
   key is also given. Note that SEARCH results aren't sorted by
   relevancy; SORT is needed for that.

5. Fuzzy Matching with Non-String Search Keys

   Fuzzy matching is not limited to just string matching. All search
   keys SHOULD be matched fuzzily, although exactly what that means for
   different search keys is left for server implementations to decide --
   including deciding that fuzzy matching is meaningless for a
   particular key, and falling back to exact matching. Some suggestions
   are given below.

Dates:
    A typical example could be when a user wants to find a message
    "from Dave about a week ago". A client could perform this search
    using SEARCH FUZZY (FROM "Dave" SINCE 21-Jan-2009 BEFORE
    24-Jan-2009). The server could return messages outside the
    specified date range, but the further away the message is, the
    lower the relevancy score.

Sizes:
    These should be handled similarly to dates. If a user wants to
    search for "about 1 MB attachments", the client could do this by
    sending SEARCH FUZZY (LARGER 900000 SMALLER 1100000). Again, the
    further away the message size is from the specified range, the
    lower the relevancy score.

Flags:
    If other search criteria match, the server could return messages
    that don't have the specified flags set, but with lower relevancy
    scores. SEARCH SUBJECT "xyz" FUZZY ANSWERED, for example, might
    be useful if the user thinks the message he is looking for has the
    ANSWERED flag set, but he isn't sure.

Unique Identifiers (UIDs), sequences, modification sequences: These
are examples of keys for which exact matching probably makes sense.
Alternatively, a server might choose, for instance, to expand a UID
range by 5% on each side.

6. Extensions to SORT and SEARCH

    If the server also advertises the SORT capability as defined by
    [SORT], the results can be sorted by the new RELEVANCY sort criteria:

        C: C1 SORT (RELEVANCY) UTF-8 FUZZY SUBJECT "Helo"
        S: * SORT 5 10 1
        S: C1 OK Sort completed.

    The message with the highest score is returned first. As with the
    RELEVANCY return option, RELEVANCY sort criteria MUST NOT be used
    unless a FUZZY search key is also given.

    If the server also advertises the ESORT capability as defined by
    [CONTEXT], the relevancy scores can be retrieved using the new
    RELEVANCY return option for SORT:

        C: C2 SORT RETURN (RELEVANCY ALL) (RELEVANCY) UTF-8 FUZZY TEXT
            "Helo"
        S: * ESEARCH (TAG "C2") ALL 5,10,1 RELEVANCY (99 42 4)
        S: C2 OK Sort completed.

Furthermore, if the server advertises the CONTEXT=SORT (or
CONTEXT=SEARCH) capability, then the client can limit the number of
returned messages to a SORT (or a SEARCH) by using the PARTIAL return
option. For example, this returns the 10 most relevant messages:

```
C: C3 SORT RETURN (PARTIAL 1:10) (RELEVANCY) UTF-8 FUZZY TEXT
   "World"
S: * ESEARCH (TAG "C3") PARTIAL (1:10 42,9,34,13,15,4,2,7,23,82)
S: C3 OK Sort completed.
```

7. Formal Syntax

The following syntax specification uses the augmented Backus-Naur
Form (BNF) as described in [ABNF]. It includes definitions from
[RFC3501], [IMAP-ABNF], and [SORT].

```
capability         =/ "SEARCH=FUZZY"

score              = 1*3DIGIT
   ;; (1 <= n <= 100)

score-list         = "(" [score *(SP score)] ")"

search-key         =/ "FUZZY" SP search-key

search-return-data =/ "RELEVANCY" SP score-list
   ;; Conforms to <search-return-data>, from [IMAP-ABNF]

search-return-opt  =/ "RELEVANCY"
   ;; Conforms to <search-return-opt>, from [IMAP-ABNF]

sort-key           =/ "RELEVANCY"
```

8. Security Considerations

Implementation of this extension might enable denial-of-service
attacks against server resources. Servers MAY limit the resources
that a single search (or a single user) may use. Additionally,
implementors should be aware of the following: Fuzzy search engines
are often complex with non-obvious disk space, memory, and/or CPU
usage patterns. Server implementors should at least test the fuzzy-
search behavior with large messages that contain very long words
and/or unique random strings. Also, very long search keys might
cause excessive memory or CPU usage.

Invalid input may also be problematic. For example, if the search
engine takes a UTF-8 stream as input, it might fail more or less
badly when illegal UTF-8 sequences are fed to it from a message whose

character set was claimed to be UTF-8. This could be avoided by
validating all the input and, for example, replacing illegal UTF-8
sequences with the Unicode replacement character (U+FFFD).

Search relevancy rankings might be susceptible to "poisoning" by
smart attackers using certain keywords or hidden markup (e.g., HTML)
in their messages to boost the rankings. This can't be fully
prevented by servers, so clients should prepare for it by at least
allowing users to see all the search results, rather than hiding
results below a certain score.

9. IANA Considerations

   IMAP4 capabilities are registered by publishing a standards track or
   IESG-approved experimental RFC. The "Internet Message Access
   Protocol (IMAP) 4 Capabilities Registry" is available from
   http://www.iana.org/.

   This document defines the SEARCH=FUZZY IMAP capability. IANA has
   added it to the registry.

10. Acknowledgements

   Alexey Melnikov, Zoltan Ordogh, Barry Leiba, Cyrus Daboo, and Dave
   Cridland have helped with this document.

11. Normative References

   [ABNF]   Crocker, D., Ed. and P. Overell, "Augmented BNF for
                Syntax Specifications: ABNF", STD 68, RFC 5234,
                January 2008.

   [CONTEXT]   Cridland, D. and C. King, "Contexts for IMAP4",
                RFC 5267, July 2008.

   [ESEARCH]   Melnikov, A. and D. Cridland, "IMAP4 Extension to SEARCH
                Command for Controlling What Kind of Information Is
                Returned", RFC 4731, November 2006.

   [IMAP-ABNF] Melnikov, A. and C. Daboo, "Collected Extensions to
                IMAP4 ABNF", RFC 4466, April 2006.

   [KEYWORDS]   Bradner, S., "Key words for use in RFCs to Indicate
                Requirement Levels", BCP 14, RFC 2119, March 1997.

   [RFC3501]   Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION
                4rev1", RFC 3501, March 2003.

   [RFC5255]   Newman, C., Gulbrandsen, A., and A. Melnikov, "Internet
                Message Access Protocol Internationalization", RFC 5255,
                June 2008.

   [SORT]   Crispin, M. and K. Murchison, "Internet Message Access
                Protocol - SORT and THREAD Extensions", RFC 5256,
                June 2008.

Author's Address

   Timo Sirainen

   EMail: tss@iki.fi