American University in Bulgaria
Peter Lazarov Lakov

The Keyword Protocol (KP)

<draft-rfced-exp-lakov-00.txt>

Status of This Memo

i) Summary

This document provides a proposal for a new Internet protocol, which should improve the relevance of results of search requests for information in the WWW. It contains a description of the basic model and the minimum set of commands necessary to run properly. It is neither meant to nor should be regarded as a definitive version, but rather as a suggestion to a new way of

looking at the relations between the search engines on one hand and hosts running WEB servers on the other.

ii) Introduction

The purpose of this protocol is to improve the results of search requests sent to search engines like Alta-Vista, Lycos, Infoseek, etc.

Whenever a user browsing the WEB sends a search request to any of the search engines, there is a high probability that the result will contain many irrelevant entries. This happens because of the way WEB wanderers, robots and spiders fill in and update their databases. Currently their algorithms for finding information is visiting different sites, browsing the file and taking out keywords which will be used in subsequent queries. The keywords taken out will depend on the algorithms of the robots, but generally they will be either too few, or too many. If robots parse only the title of a file, then surely many keywords, useful for finding the file, will not be available, for it is not possible to include them all in the title. If, on the other hand, the whole file is parsed, surely many keywords, which have nothing to do with the topic of the file, will be included, thus enabling this file to appear in search results for completely different topics. In any case, the owner of the file (or the WEB server supervisor) - the main person interested in assuring that the information will reach the target audience - has no power in assisting this process. His/her actions are reduced to the very passive role of only providing access to the data.

The HTML tags contents and keywords can only partially alleviate this problem, because many of the HTML files, linked in a document already parsed by a robot, do not contain those tags. In this case the robot should decide whether keywords should be extracted from those files, or they should be disregarded. A search engine may extracts keywords from a file which was not intended to provide such - in this case it will fill its database with worthless data. Similarly, it may neglect files with some important data. In either cases, the owner of the files has very little power in communicating the precise information to the robots.

The KP protocol will give the WEB server supervisor and each single user an opportunity for very close control of the information they provide for public access. Each user will be able to edit the exact keywords necessary to describe his/her files. The cornerstone of the suggestion is that all descriptions of the files will be handled by a central and well-known server, which will both increase accuracy and decrease the time necessary to browse a WEB server.

iii) Overview of the job done by the KP server.

The KP server operates on the client-server paradigm through a reliable TCP/IP byte stream using the ASCII character set. The server performs a listen on a well-known port and when a client requests a connection to that port, the server accepts the connection. Once it is created, the client starts sending commands to the server, which performs the action and returns a response and (when applicable) data. The response of the server may be of either predictable, or unpredictable line length. In case the an unpredictable line length answer, the last line contains only a full stop. Commands and replies are terminated by a new line character (more on the command syntax in part 4.)

>From the point of view of the server, there are two types of objects which can contact the server - users and robots (from now on, till the end of the document, I shall refer to a robot as program written only to contact the KP server and update the databases of search engines. Do not confuse with previously mentioned robots, web wanders, crawlers.) The difference between the two is that first, users need to supply password, while the robot does not. Actually, the robot does not present any identification whatsoever, so any person without a user permission could login as a robot. There is no harm taken in this, because the information is meant for public use anyway. Second, users have permission to edit some of the information, while robots have only read only permissions.

The KP server needs to handle a single copy of each of following files with the following proposed fields:

File:	DATA
Fields:	<id>, <user>, <file>, <keywords>

File:	PASSWORD
Fields:	<user>, <password>

File:	PATCH
Fields:	<patch_N>, <patch_N+1 >

File:	USED_PATCH
Fields:	<patch_file>

and many of the following tables:

File: P_1, P_2, P_3, , P_N
Fields: <id>, <action>, <file>, <keywords>

where the meaning of the fields for each table is as follows:

DATA

<id>	a unique record identifier (same for the other tables)
<user>	the name of the user, to whom the file belongs
<file>	the actual name of the file
<keywords>	a string with keywords, separated by comma

PASSWORD

<user>	same as in table DATA
<password>	the password for <user>, used at login time.

PATCH

<patch_N>	the name of the previous to last patch file
<patch_N+1>	the name of the last patch file

USED

<patch_file>	the name of the patch files, already sent to robots.

P_1, P_2, P_N

<id>	same as <id> in DATA
<action>	action to be taken when the patch file is merged into the database. N is for new (add new record with <id>, <file> and <keywords> like those in the current table), D is for delete (Delete the records with <id> equal to <id> of the current table)
<file>	same as <file> in DATA
<keywords>	same as <keywords> in DATA

File DATA contains information concerning all available files and their respective keywords. When a robot contacts KP server for the first time, it should first download the file DATA using the GETALL command (which sends back all the records of file DATA.

Then, the robot can send the command NEXTPATCH a number of times until it records all the changes done to file DATA. The rule for generating a new patch file is simple: whenever a robot visits the last patch file, create a new patch file and use it to store all changes thereafter.

Changes are made only by users (see above,) only with the commands ADDFILE and DELETEFILE. Whenever one of these two commands is used, the action taken is stored to the last, unvisited by robots, patch file. Each user can change only the files, which are referred to by DATA<user> as his/her username.

iv) Communication with the server.

Each command should be terminated with CRLF characters. The space left blank between the commands and the parameters should be considered as white space. CRLF characters and white spaces are not shown explicitely in the description of the commands lest they become too overburdened.

iv.i) Communication with remote computers (wanders, robots.)

GETALL	the server sends to the client all the records with the files and keywords. ACTION is N for all the records.
GETPATCH <patchname>	the server sends to the client only the records from the file <patchname>.
NEXTPATCH <patchname>	the server sends to the client only the name of the next patch file. No records from the patch are actually transferred. If <patchname> is empty, then the return value is the first patch of the whole database.

iv.ii) Communication with local computers (those updating the files.)

USER <username>	the client sends to the server the username of the person who wants to update the database. Username robot is reserved for robots, WEB wanderers and staff.
PASS <password>	the client sends to the server the password of the user.
ADDFILE <filename, info>	the client sends to the server

	a line containing a filename (possibly the URL) and the keywords which should get in the search engines databases for that file. In case there is already an entry in the server s database for that file the keywords should be replaced with the new ones.
DELETEFILE <filename>	the server deletes the entry for this file from its database. A user would typically want to do this operation if the file is deleted or moved to a new position. If the last patch file has been sent to at least one robot/wander (or there are no patch files yet), the server should create a new patch file and add the entry in it.
LISTLIKE <pattern>	the server sends to the client a list of files matching the specified condition. If the <pattern> is empty, the server sends all the files
LISTMINE	the server sends to the client only the files belonging to the user currently logged in.
EXACT	switch exact string comparison ON/OFF. When exact mode is ON, a string is equal to another only when they have the same sequence of characters. When exact is OFF, a string is equal to another when it is a sub-string of the second. All comparison is case-sensitive. When exact mode is OFF,
HELP <command>	the server sends a short help message to the client about the command specified. If no command is specified, the server sends the list of all the commands.
QUIT	request that the connection with the server be terminated.

v) Actions taken for each command.

USER <username>

1) Check if username is robot . If yes, then this is a robot.
Let it in without asking for password and apply only the commands
for robots. If it enters other commands, then send a message 205.

2) If the name is not "robot", check for it in the table
password. If the name is not found, send a message 210. Else
send message 101.

PASS <password>

1) Check whether user has already logged in. If yes, send a
message 204.
2) If the user hasn t logged in yet, check the password sent
against the one stored in file password for that user. If
different, send 207. Else send 102.

ADDFILE <filename, info>

```
1) Add the record to table DATA.
2) Send message 103.
3) Check if there is already a patch file.
        * If no patch file exists yet,
                * add field to table PATCH with fields: DATA, P_1.
                * Create patch file P_1 and add the field to it.
        * If a patch file exists,
                *locate the last patch
                *If it has been sent to robots,
                        * Add field to table PATCH with: P<prev>, P<prev + 1>
                        *Create patch file P<prev+1> and add the field to it.
                *If it hasn t been sent to robots:
                        * Add the field to the last patch file.
4) Send confirmation message 103
```

DELETEFILE <filename>

1) Locate the file and check that it belongs to the user. If the
file is not present in the database, send message 208. If the
file does not belong to the user, send message 209. In either of
the two cases, goto step 6)
2) Delete the record from table DATA.
3) Send message 104.
4) Follow the same steps, as for ADDFILE step 3.
5) Send confirmation message.
6) End of DELETEFILE command.

LISTLIKE <pattern>

1) Send the user the files matching the pattern.

LISTMINE

1) Send the user only the files belonging to him/her.

EXACT

1) Change the comparison mode.
2) Send the user the message 105 or 106.

HELP <command>

1) Send to the user a help for the command. If <command> is empty, send a list of all available commands.

GETALL

1) Send all records of file DATA.

GETPATCH <patchname>

1) Send all records of file <patchname>.

NEXTPATCH <patchname>

2) Locate for next patch file in PATCH table.
3) Send message 107.
4) Send name of next patch file.

vi) Messages.

101	**+OK Enter password.**
102	**+OK Welcome to KP version 1.0.**
103	**+OK Your file has been added to the database.**
104	**+OK The file has been deleted from the database.**
105	**+OK Exact mode is ON.**
106	**+OK Exact mode is OFF.**
107	**+OK The next patch file is:**
201	**-ERR Unknown command.**
202	**-ERR Command USER expected.**

203	**-ERR Command PASS expected.**
204	**-ERR You have already logged in.**
205	**-ERR Command not allowed for your class.**
206	**-ERR No patch file with this name.**
207	**-ERR Password incorrect. Try again.**
208	**-ERR File ID not found.**
209	**-ERR You have no write permission for this file.**
210	**-ERR User unknown.**

Author's Contact Informationa

Peter Lakov
lakov@wizcom.bg