lib_mysqludf_ssdeep Documentation

Release 1.0.1

Simon Holywell

Sep 27, 2017

Contents

ssdeep is a utility for creating and comparing fuzzy hashes or context-triggered piecewise hashing.

Fuzzy hashing can match signatures which have "...sequences of identical bytes in the same order, although bytes in between these sequences may be different in both content and length". (ssdeep project page)

This MySQL User Defined Function provides functions for creating and comparing fuzzy hashes.

Contents:

CHAPTER 1

Installation

Get the source

Either clone down the git respository from github or download the latest tag as a tar gzip:

```
$ git clone git://github.com/treffynnon/lib_mysqludf_ssdeep.git
```

This should beplaintext yeah

Install dependencies

Ubuntu/Debian

To install the MySQL development headers and ssdeep install the following package:

```
$ sudo apt-get install libmysqlclient-dev ssdeep libfuzzy-dev
```

Redhat

To install the GCC C++ compiler and the MySQL development headers install the following packages:

```
$ yum install gcc-c++ mysql-devel autoconf automake libtool
```

As there is no libfuzzy package on Redhat you need to build ssdeep from its sources:

```
$ cd ssdeep
$ ./configure
$ make
$ make install
```

Build lib_mysqludf_ssdeep

Now onto building the lib_mysqludf_sseep library:

```
$ cd lib_mysqludf_ssdeep
$ ./build.sh
```

The library file, lib_mysqludf_ssdeep.so, is now in the current directory ready to be installed into MySQL,

Install lib_mysqludf_ssdeep

The library needs to be copied to the MySQL plugin directory. To find out the location of your plugin directory run the following SQL

```
show variables like 'plugin_dir';
```

A common location is /usr/lib/mysql/plugin, but it can be changed in the MySQL configuration at /etc/mysql/my.cnf in the mysqld section:

```
[mysqld]
plugin_dir=/usr/lib/mysql/plugin
```

To install the library execute the following commands:

```
$ sudo cp lib_mysqludf_ssdeep.so /usr/lib/mysql/plugin/
$ mysql -u root -p < src/installdb.sql
```

CHAPTER 2

Quickstart Tutorial

This UDF is quite simple and its use case is really very clear, but for the impatient many here is a lightening quick run down to get a project up and running.

So if you are writing a system that has to store a large number of documents, but you don't want duplicates or near duplicates cluttering up the application then you can employ ssdeep to generate hashes for you. These hashes can then be compared to any future documents that maybe uploaded to determine their likeness.

Firstly create a documents table:

```
CREATE TABLE IF NOT EXISTS `documents` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `ssdeep_hash` text COLLATE utf8_unicode_ci NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Next up create two text files and save them as /tmp/document_1.txt and /tmp/document_2.txt. A simple way of getting text is just to nab a few articles from a news website or wikipedia.

> **Warning:** These files should contain at least 4 kilobytes of text for an accurate hash to be created. The is a restriction of the fuzzy hashing algorithm itself and not lib_mysqludf_ssdeep.
>
> I have had success with hashes taken from text with just 250 characters though so your mileage may vary.

Then "upload" the first document with the following insert statement:

```
INSERT INTO `documents`
SET `title` = 'The first document!',
    `ssdeep_hash` = ssdeep_fuzzy_hash_filename('/tmp/document_1.txt');
```

So when another document is uploaded you can compare its signature against the documents already in the documents table.

```
SELECT `title`
FROM `documents`
WHERE ssdeep_fuzzy_compare(`ssdeep_hash`, ssdeep_fuzzy_hash_filename('/tmp/document_2.
↪txt')) > 20;
```

Every document that is returned has more than a 20% likeness to the freshly uploaded document.

CHAPTER 3

Function Reference

ssdeep_fuzzy_hash

ssdeep_fuzzy_hash(*to_hash*)

Calculates an ssdeep fuzzy hash for the supplied string.

Parameters **to_hash** (*string*) – The string to create a hash from - should be 4kb or greater in size

Return type string

> **Warning:** These files should contain at least 4 kilobytes of text for an accurate hash to be created. The is a restriction of the fuzzy hashing algorithm itself and not lib_mysqludf_ssdeep.
>
> I have had success with hashes taken from text with just 250 characters though so your mileage may vary.

Example

```
SELECT ssdeep_fuzzy_hash('A 4kb string would go here.');
```

ssdeep_fuzzy_hash_filename

ssdeep_fuzzy_hash_filename(*file_name*)

Calculates an ssdeep fuzzy hash from the supplied file path.

Parameters **file_name** (*string*) – The path to a file on disk to create a hash from

Return type string

Warning: These files should contain at least 4 kilobytes of text for an accurate hash to be created. The is a restriction of the fuzzy hashing algorithm itself and not lib_mysqludf_ssdeep.

I have had success with hashes taken from text with just 250 characters though so your mileage may vary.

Example

```
SELECT ssdeep_fuzzy_hash_filename('/tmp/file.txt');
```

ssdeep_fuzzy_compare

ssdeep_fuzzy_compare(*signature1*, *signature2*)

Calculates match percentage between two hash strings.

Parameters

- **signature1** (*string*) – ssdeep hash to compare
- **signature2** (*string*) – ssdeep hash to compare

Returns 0 to 100 on success and null otherwise.

Return type integer or null

Example

```
SELECT ssdeep_fuzzy_compare('384:eGWhC3Uvw60bolB5EqQWXVyBp5ZXHqDj2Gub:mYUvywB620rZPh',
↪ '48:H46piMWRaKuCghtYCzHq2nzu50mAmyyGl7A/
↪eIMTQySmmf2ysIX3zxPZ:H46I4tYCb9xmyyGlsMT9Smg2ysUZ');
```

lib_mysqludf_ssdeep_info

lib_mysqludf_ssdeep_info()

This function will return the version number of the UDF.

Return type string

Example

```
SELECT lib_mysqludf_ssdeep_info();
```

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Index

L

S