SiteMind Documentation

Release latest

May 17, 2018

Contents

Domain research tool targeting media planners and reseaerchers, specifically built for countering ad fraud and reducing its impact on media investment. Returns a result with up to 40 signals for any site typically in 1-3 seconds.

CHAPTER 1

FEATURE HIGHLIGHTS

- intuitive ‘buy score’ system for website rating
- search any domain
- returns result usually in 1 to 2 seconds
- up to 150 data points per site from 5 different sources
- easy to use API with ready end-points for all common languages

CHAPTER 2

OVERVIEW OF FUNCTION

SiteMind allows two different kinds of searches to be performed by the user:

- type-1: where a single domain name is the input
- type-2: where a comma separated list of domain names is the input

In both cases the system performs a series of operations resulting in up to 40 signals, which are then stored in a .csv file. Depending on the type of search, the result will then be returned either as a simple user interface, or a table with results for multiple sites.

CHAPTER 3

SITEMIND SCORING SYSTEM

The SiteMind scoring system takes widely accepted “red flags” from signals available from various data sources (see the sections below) and creates a single easy to understand score out of those flags.

The formula to calculate the score ranging from 0 to 100 is as follows:

100 - ((CHECKS FAILED / CHECKS TOTAL) * 100) = SiteMind SCORE

The score consist of 10 “flags”.

VARIABLE NAME	FAILS WHEN
SCORE_CHECKS	Not enough signals to perform 4 checks
SCORE_UPSTREAM	More than 90% of the traffic coming from TOP5 Upstream
SCORE_UPSTREAMCHECK	No common sites in TOP5 Upstream
SCORE_TRUST	Web of Trust Trust score is less than 50
SCORE_TOPKEYWORDS	More than 90% of the traffic coming from TOP5 Keywords
SCORE_SEARCH	Less than 1% of traffic is coming from search
SCORE_PAGEVIEWS	More than 8 pageviews per visit on average
SCORE_YEARS	Domain was originally registered less than 2 years ago
SCORE_PRIVACY	Domain uses whois privacy guard
SCORE_BOUNCERATE	Site bouncerate is less than 10% on average

CHAPTER 4

DATA TAXONOMY

The below table shows all the signals that are currently available through SiteMind. All variables are available through the scan function in the resulting .csv file, or in the user interface resulting from a single site search.

NOTE: Different naming may be used in the user interfaces, and this is easily changed.

VARIABLE NAME is the name of the variable as it is found in the output file resulting from a search of scan.

SOURCE is the reference to where the data is originating from. In the case the filed says 'sitemind' it means that the signal is inferred from other data.

COLUMN NUMBER is only for development purpose and is used in the UI codes to to present a certain signal in a given place in the user interface.

VARIABLE NAME	SOURCE	COLUMN NUMBER
SCORE_CHECKS	sitemind	2
SCORE_UPSTREAM	sitemind	3
SCORE_UPSTREAMCHECK	sitemind	4
SCORE_TRUST	sitemind	5
SCORE_TOPKEYWORDS	sitemind	6
SCORE_SEARCH	sitemind	7
SCORE_PAGEVIEWS	sitemind	8
SCORE_YEARS	sitemind	9
SCORE_PRIVACY	sitemind	10
SCORE_BOUNCERATE	sitemind	11
ADMIN_CITY	whois	12
ADMIN_COUNTRY	whois	13
ALEXA_BOUNCERATE	alexa	14
ALEXA_INLINKS	alexa	15
ALEXA_LOADSPEED	alexa	16
ALEXA_PAGEVIEWS	alexa	17
ALEXA_RANK	alexa	18
ALEXA_SEARCHVISITS	alexa	19

Continued on next page

Table 1 – continued from previous page

VARIABLE NAME	SOURCE	COLUMN NUMBER
ALEXA_TIMEONSITE	alexa	20
ALEXA_TOPCOUNTRIES	alexa	21
ALEXA_TOPKEYWORDS	alexa	22
ALEXA_UPSTREAM1	alexa	23
ALEXA_UPSTREAM1N	alexa	24
ALEXA_UPSTREAM2	alexa	25
ALEXA_UPSTREAM2N	alexa	26
ALEXA_UPSTREAM3	alexa	27
ALEXA_UPSTREAM3N	alexa	28
ALEXA_UPSTREAM4	alexa	29
ALEXA_UPSTREAM4N	alexa	30
ALEXA_UPSTREAM5	alexa	31
ALEXA_UPSTREAM5N	alexa	32
CHECKS_TOTAL	alexa	33
CHECK_FALSE	alexa	34
CHECK_TRUE	alexa	35
TOP5_UPSTREAM	alexa	36
WHOIS_PRIVACY	whois	37
WHOIS_YEARS	whois	38
WOT_CHILDSAFETY	weboftrust	39
WOT_TRUST	weboftrust	40

CHAPTER 5

DATA SOURCES

While adding virtually any additional data soruce, SiteMind relies on three different data source by default.

- Alexa
- Web of Trust
- WHOIS

ALEXA*

It is recommended to use the paid Alexa API. SiteMind uses web scraping method by default for demo and prototyping purpose.

Web of Trust

Web of Trust data is fetched using the WOT API, which provides a rich data taxonomy and is free to use to a substantial level of daily usage.

More information on the WOT API can be found here: https://www.mywot.com/wiki/API

You can apply for your own API key here: https://www.mywot.com/en/reputation-api

WHOIS

SiteMind provides a fully automated method for the "gold standard" way of fetching WHOIS records.

1. Gets to main record from the tld level registar including the registar that holds the sub-record
2. Gets the sub-record from the holding registar

CHAPTER 6

PROCESS FLOW

1. **User provides input through the search field in the UI**
    - > form_process.php
    - > run.sh
2. **run.sh checks if there query is empty, single domain, or multiple comma separated domains**
    - > sitemind.sh (“controller”)
3. **Regardless if it’s single or multi search the program cycle proceeds**
    - **> bin/api-fetch.sh**
        - > bin/alexa_data.sh
        - > bin/whois_data.sh
        - **> bin/wot_data.sh**
            - > wo_data.py
4. **Using the data in various .temp and .bash files a usable data format is created**
    - > bin/api-build.sh
5. **The data is provided in a comma separated format for multi searches**
    - > data-export.sh
6. **The data is further formatted for the UI building process**
    - > data-cms.sh
7. **The UIs are built each in a separate script**
    - > cms/cms-scorecard.sh
    - > cms/cms-traffic.sh
    - > cms/cms-overview.sh
    - > cms/cms-upstream.sh

8. **A finish cleanup is performed**
    - > finish-cleanup.sh

CHAPTER 7

DIRECTORY STRUCTURE

NOTE: In a multi-user system, each user has a self-contained replica of the program folder in the program root.

FOLDER	
/	Sitemind program root
/bin	Where non UI scripts reside
/cms	Where the UI scripts reside
/cms/graphics	Images for the UI
/cms/js	Javascripts for the UI
/cms/style	Style sheets for the UI
/cms/templates	Header and Footer for UI

CHAPTER 8

GETTING STARTED

The following installation instructions have been tested on Ubuntu 16.04 clean distro.

Install dependencies:

```
sudo apt-get update
sudo apt-get install -y apache2
sudo apt-get install -y php5
sudo apt-get install -y unzip
sudo apt-get install -y parallel
sudo apt-get install -y num-utils
sudo apt-get install -y git
```

Getting the source files and setting it up:

```
wget https://github.com/SiteMindOpen/SiteMind/archive/master.zip
unzip master.zip
sudo rsync -av ~/SiteMind-master/ /var/www/html

chown -R www-data:www-data /var/www/html && chmod -R g+rw /var/www/html
```

After the initial setup, as long as you create new users with SiteMind command line command 'sm-user-new', permissions will be handled automatically and is not something you need to think about.

Creating an admin user:

```
PASSWORD=$(openssl rand -base64 20); htpasswd ./etc/apache2/.htpasswd -cbB admin "
↪$PASSWORD"; echo -e "Your password is $PASSWORD";
```

Restart apache

Ubuntu 14.04:

```
service apache2 restart
```

Ubuntu 16.04:

```
systemctl apache restart
```

8.1 HTTPS WITH LETSENCRYPT

Letencrypt makes it incredibly easy (and fast) to setup functional https for your site.

Note that for the below to work, you need to have a valid domain name that is pointed to the server you're initiating the below command from:

```
sudo git clone https://github.com/letsencrypt/letsencrypt /opt/letsencrypt
cd /opt/letsencrypt
./letsencrypt-auto --apache -d yoursite.com
```

NOTE: as part of the setup process, there will be a prompt asking if you want to redirect all requests to https. I think this should be on for most cases.

CHAPTER 9

DEBUGGING

For **DATA related** debugging change production_version to debug_function from line 24 in bin/api-fetch.sh. This will help you to identify issues with one part of the data fetching cycle getting stuck. This should happen very rarely as it has been debugged a lot.

For **UI related** download the program folder to a local machine and run a PHP server locally. This way you will very easily see any error messages that are coming up when the UI is loaded.

If you've setup properly, then you can easily see related error logs on the server-side using:

```
./sm-monitor
```

CHAPTER 10

RUNNING LOCALLY

You have to run a PHP server from the Sitemind folder to be able to make queries from the UI:

```
php -s http://127.0.0.1:8000
```

If you're a mac user, go the Sitemind folder and exexcute the below command:

```
sudo php -S 127.0.0.1:8000 && /Applications/Google\ Chrome.app/Contents/MacOS/Google\␣
↪Chrome --app="http://127.0.0.1:8000/dev/index.html" --window-size="1000x800"
```

Alternatively you can run from the command line (in the Sitemind folder):

```
./run.sh domain.com
```

CHAPTER 11

CODING CONVENTIONS

The code is almost 100% bash and certain principles have been followed where possible:

- code starts one tab intend deep
- each script (.sh file) represents a step in the process flow
- no more than 50 lines of code per script
- no more than 50 characters long lines of code
- functions first, program second, cleanup last
- mininal comments - instead self-explaining code

It should be very easy for anyone with beginner+ level in bash to modify the code that is already there, to add new code to improve current functionality, or add completely new functionality.

CHAPTER 12

FUTURE DEVELOPMENT

- Create setup process where server is configured including SSL and a conf file is created at ~/.sitemindrc
- Make upstream sites clickable (yields a new search)
- Check for native advertising being a major source of traffic
- Add a 30 day cache to avoid redundant searches
- Make one-page report for export available with all the signals
- time-limited account creation

CHAPTER 13

ADMIN FEATURES

In the environment of the host machine, include the following alias commands:

```
alias sm-sync='/var/www/html/admin/bin/sync.sh'
alias sm-user-list='cat /etc/apache2/.htpasswd | cut -d: -f1'
alias sm-monitor='/var/www/html/admin/bin/monitor.sh'
alias sm-user-new='/var/www/html/admin/bin/user-new.sh'
alias sm-user-rm='/var/www/html/admin/bin/user-sh.sh'
alias sm-commit='/var/www/html/admin/bin/commit.sh'
alias sm-commit-version='cd ~/git/sitemind && /var/www/html/admin/bin/commit-version.
↪sh'
alias sm-commit-log='git log --oneline --decorate --color'
alias sm-conf-nossl='vim /etc/apache2/sites-available/000-default.conf'
alias sm-conf-ssl='vim /etc/apache2/sites-available/000-default-le-ssl.conf'
alias sm-find-file='/var/www/html/admin/bin/sm-find-file.sh'
```

Usually you can find the file from ~/ under the name .bashrc. Add the above lines in to the file and next time you login to the host, the following commands will be available anywhere in your system:

In a Linux system you can do this typically by:

```
vim .bashrc
```

sm-sync

Syncs all the user accounts with /dev.

sm-user-list

Prints out a list of user accounts.

sm-monitor

Creates a report out of access and error logs from the on going day's logs.

sm-user-new

Creates a new user in to the system and prints out a randomly generated password for the user.

EXAMPLE USAGE (where we want to create a user 'john':

```
sm-newuser john
```

sm-user-rm

Removes a user and all associated files from the system (Use with caution!).

13.1 SERVER CONFIGURATION

sm-conf-nossl

Opens up the no ssl (port 80) apache configuration file in vim editor.

sm-conf-ssl

Opens up the ssl (port 443) apache configuration file in vim editor.