lookuper

lookuper performs lookups against VirusTotal/ThreatExpert/Google Safe Browsing data for data types such as MD5, SHA256, URL, IP, Domains, Strings e.g. mutexes

Data is cached in a local SQLite database. The VirusTotal functionality supports multiple API keys, so you can supply four public API keys and it will run continuously, iterating through the API keys.

Features

- Data caching in SQLite database
- Support for multiple API keys
- Batched requesting
- Supports hashes (MD5 & SHA256), URL's, IP's and domains from VirusTotal
- Supports hashes (MD5, strings) from ThreatExpert
- Supports domains/URL's from Google SafeBrowsing

Configuration

The applications configuration is read from the **lookuper.config** file located in the same directory as the binary.

The configuration holds the following values:

- safe_browsing_api_key: The API key used for the Google Safe Browsing functionality
- virus_total_api_keys: The API key(s) used for the VirusTotal

functionality
- max_hash_age: The maximium age that the data is held before being deemed as stale

Example

The following is an example layout of the **lookuper.config** file:

```
safe_browsing_api_key: ABC....
virus_total_api_keys:
- AAA...
- BBB...
- CCC...
- DDD...
max_hash_age: 30
```

Command Line

The application uses commands to perform work for the different data types:

```
NAME:
   lookuper - Looks stuff up...

USAGE:
   lookuper [global options] command [command options] [argur

VERSION:
   0.0.1

AUTHOR(S):
   Mark Woan <markwoan@gmail.com>
```

```
COMMANDS:
     resume    Resumes an existing process
     clear     Clears the work queue
     md5vt     Check MD5 hashes via VirusTotal
     md5te     Check MD5 hashes via ThreatExpert
     sha256vt  Check SHA256 hashes via VirusTotal
     ipvt      Check IP addresses via VirusTotal
     domainvt  Check domains via VirusTotal
     urlvt     Check URL's via VirusTotal
     stringte  Check strings via ThreatExpert
     gsb       Check Url's/Domains via Google Safe Browsing
     help, h   Shows a list of commands or help for one comma

GLOBAL OPTIONS:
   --help, -h     show help
   --version, -v  print the version
```

An example command line and output is shown below:

```
./lookuper md5vt -i md5-28.txt -o .
2016/09/06 15:18:23 Data type: MD5 (VT)
2016/09/06 15:18:23 Loading data
2016/09/06 15:18:24 Loaded No. items: 28
2016/09/06 15:18:24 Data type: MD5 (VT)
2016/09/06 15:18:24 API key: ABC.......
2016/09/06 15:18:24 Batch size: 4
2016/09/06 15:20:38 Complete
2016/09/06 15:20:38 Cache hits: 0
```

Compilation

This document assumes that the **golang** tool set has been installed.

Golang applications always have a **src** directory which contains the

applications source code, along with any associated projects that are referenced by the primary application. The following shows where the applications source code resides:

```
/lookuper/source/src/woanware
```

gb

The project uses gb for building the project. **gb** allows for reproducible builds and vendoring so that all dependencies are kept with the project source.

Compile with gb

To compile the application use the following commands (assuming the same directory structure):

```
$ cd /lookuper/source
$ gb build all
```

Compile under Windows

The go-sqlite3 database driver used to access the Sqlite database is a cgo package, therefore gcc is needed to build the application. On linux this is generally not an issue as most **normal** people :-) have gcc installed.

On Windows, that is not always the case. So to build on Windows (x64), perform the following steps:

- Download tdm-gcc (x64) and install. Accept the defaults.

- Open a command prompt and CD to:

  ```
  C:\TDM-GCC-64
  ```

- Execute the **mingwvars.bat** file which will get the correct environment variables
- Then CD to the lookuper source code and follow the same instructions for compiling using **gb**