WotWizard Manual

WotWizard builds, from the blockchain and sandbox of Duniter, a prediction of the future entries of candidates into the Duniter Web of Trust (WOT). It uses a simulation of the Duniter mechanism. When several possibilities may happen, each one is listed with its probability. The published lists are automatically updated every five minutes, and any change is signaled visually.
WotWizard contains several tools and a fast **Web of Trust Explorer**.

Warning: You must use the 1.7.17 version of duniter or any later version in the 1.7.x series.

How to use it?

This program needs a **Duniter** node running on the same computer.

This node must be configured for the use of WotWizard. To do so, edit its configuration file ("conf.json"; see *Configuration* below) in a text editor and put the value of the "storage.wotwizard" field to "true".

WotWizard includes a server executable (**wwServer**) written in *Go* (v1.13.6) and running under GNU / Linux. This server communicates by the way of disk files containing *GraphQL* requests on input, and JSON answers on output. WotWizard includes an optional graphical user interface too (**WotWizard.exe**) written in *Component Pascal* (BlackBox v1.7.1, running under Wine). This GUI creates GraphQL requests through menus items and interactive windows, and displays answers through texts or graphics.

wwServer is to be placed in an empty directory where it can be run. It creates itself the sub-directories and files it needs, all of them being included in the inital directory. If the Duniter database is not at its standard place, you can give it with a command line option.

WotWizard.exe runs natively under Windows. In order to make it work with wwServer, place it in the same directory. Once *wine* installed, you can run it.

Server Mode

You can use WotWizard as a server only, if you want to display its results yourself, e.g. in a web page. Configure first Duniter for WotWizard (see above). If Duniter installation is standard, (its database must lie in the file "~/.config/duniter/duniter_default/wotwizard-export.db"), you can run wwServer without options:

```
$./wwServer
```

Otherwise, on the first run, add the -du option:

```
$./wwServer -du <path to Duniter database>
```

On later runs, the -du option is no more useful.

It's also possible to modify this path later by editing the "rsrc/duniter/init.txt" file.

The amount of memory allocated by wwServer for the calculus of its predictions is limited. Its default value is 430,000,000 bytes. You can change it by editing the "rsrc/duniter/parameters.txt" file. The larger it is, the more precise are WotWizard predictions, but if it's too big, wwServer may crash.

Any change in "init.txt" or "parameters.txt" is taken into account only when wwServer (re)starts.

On the first run, after a not-so-long initialization phase (which starts after the recording of a new block in Duniter) that you can follow in the "rsrc/duniter/log.txt" file, request files can be deposited into the "rsrc/duniter/Work/Query" sub-directory. Their names are meaningless and you can deposit as many of them as you want. They are immediately read and destroyed. After their handling, answer files appear in "rsrc/duniter/Work/Json" sub-directory; see below the rules between requests and answers. In the case of errors in request files, error messages appear in log.txt and in the "errors" field of the json file.

To stop wwServer, press Ctrl-C or use any other standard method.

The WotWizard database is contained in the "rsrc/duniter/System/DBase.data" file. At every start of wwServer, a new copy of DBase.data is created. At most, two copies are saved: "Dbase.data.bak" is the last one and "DBase.data.bak1" is the previous one. If you have a doubt on the integrity of the datebase, replace it, once wwServer is stopped, by renaming an older copy.

List of GraphQL requests, and their meanings

The content of request files must follow the graphQL specifications of june 2018:

https://spec.graphql.org/June2018/

The graphQL types definitions document for WotWizard is available in the Help/TypeSystem.txt file.

Each request file may contain several operations (and fragments too), which are all run. For each operation, a json file is created, whose name derives from the operation name.

If the request file begins with a comment (#) on the first column, this comment must start with two hexadecimal digits, followed by a json object ({...}). The fields of this object define the variable values used by operations in the file. The number ('jsonNum') expressed by the two digits is embedded in the names of answer files (if no 'jsonNum' appears in the file, its value is supposed to be zero).

The name of an answer json file is the concatenation of the operation name, a dash (-) and the 'jsonNum' number (in decimal). If no operation could be read, its name is replaced by "-Errors". If only one anonymous operation is present, its name is replaced by "-Result".

Example:

#2A{"a":12,"b":5}

defines the two variable values $a = 12 and $b = 5, and fixes 'jsonNum' as 0x2A = 42. If an operation named "Test" appears later in the same file, the corresponding json file will be named "Test-42.json", "-Result-42.json" If the only operation is anonymous, and "-Errors-42.json" if no operation could be read.

Interactive Mode

Place the "WotWizard.exe" file in the same directory as wwServer. Run WotWizard by a double-click or with the command line (from the same directory):

```
$wine WotWizard.exe
```

WotWizard.exe doesn't start wwServer, that you must run before any request. It doesn't stop it either.

Once WotWizard has started, choose your language and, possibly, your preferred fonts, in "Edit -> Preferences...".

After that, you can run commands with menus items and see answers in opening windows.

The WotWizard Window

Open the WotWizard Window with "WotWizard -> New WotWizard View".

You can:
- choose the way the list is displayed (by names or by dates, or metadata -- analog to the 'File' data: see below)
- manually update the list (it's automatically updated every time a new block appears)

When the list has changed, two asterisks appear, one on each side of the title, and a new button "Check" is created. Click on the button to make the marks disappear. You can then compare the new and old lists (by Dates and Metadata) by clicking on the button "Compare", or by using the menu item "Edit -> Compare Texts" (keyboard shortcut: F9).

File: WotWizard calculates its forecasts from a state of affairs called *File*. It's a list of *dossiers* (newcomer alias + certifications list) and *internal certifications* (between members), sorted by availability dates. The command "WotWizard -> File" displays the following details:

- *Internal certification* : one line with:
  + the receiver and sender of the certification (aliases) with an arrow (←) between them;
  + the certification availability date;
  + between brackets, the validity limit of the certification;
- *Dossier* :

+ a first line describing the newcomer:
  * the number of main certifications (those which fix the availability date of the dossier, see below);
  * her alias;
  * between brackets:
    - her availability date;
    - its minimal availability date, two months (sigReplay) after her previous membership application;
  * between brackets, the validity limit of her application;
  * the distance rule result, as the percentage of reachable sentries;
  * a summary of the dossier state: OK if it's valid and if the rule of minimal certification number and the distance rule are verified, or KO otherwise (certification availabilities are not checked here).
+ one additional line for each external certification, in the same form as for internal certifications, but without the sender alias (which is the newcomer's); these certifications are sorted by availability date.

Main certifications: In a Dossier, it's the *n* first external certifications (in the order of availability dates), where *n* is the number of certifications it owns when it is less than or equal to *sigQty* (5). If it is greater, it's the number of certifications needed to satisfy the distance rule, or the total number if the distance rule can't be satisfied.

Permutations : The command "WotWizard -> Permutations" displays all entry ranks permutations predicted by WotWizard, with their probabilities. **Warning**: their number may be sometimes very big and the display very large!

Web of Trust Explorer

Open the WoT Explorer with the menu command "Web of Trust -> Explorer".

See the explorer map.

You can search any identity in the blockchain or the sandbox by typing its first characters or the first characters of its public key into the first top field and by clicking on the "Search" button. The possible identities appear in the list at the bottom: choose the one you want to look at. Some details appear in the "Identity" frame, and the certifications, received and sent, in the "Certifications" frame. You can look at the identity of one of the senders or receivers by clicking on the corresponding "Go" button; step back with the arrow buttons. You can also display a text of informations on certifiers or certified identities by clicking on one of the "D" buttons (list of present certifiers or certified identities, sorted by alphabetic order, by expiration dates of membership (or limit date before revocation if the

identity is no more member - "×" before her name) and by expiration dates of certifications, and list of all non-revoked certifiers or certified identities of all times, able to certify or be certified again). Names of certifiers and certified waiting in sandbox are preceded by the symbol º.
Displayed dates are member's registration date into the blockchain, expiration dates, of member's membership and of her certifications, and the availability date of the next sent certification in the field "Availability" (if already available, this date comes after an exclamation mark "!"). If member's end of membership has been requested, her expiration date of membership is preceded by a cross (×). In this case, she can no more get new certifications, and, without renewal of her membership, she will be excluded at this date.

The degree of centrality *c* of a member is the number of oriented paths (certifier -> certified) of shortest lengths to which she belongs and whose length is, at most, *stepMax* (5) plus one steps from the certifier. The centrality level *c'* of a member is calculated from her degree of centrality *c* by the expression:

$$c' = 100 \frac{\ln(1+c)}{\max[\ln(1+c)]}$$

where $\max[\ln(1+c)]$ is the greatest value of $\ln(1+c)$ for all members, and where ln is the natural logarithm.

One may consider the *centrality level* of a member as the level of her involvement to help people she certified to respect the distance rule, independently of her capacity to do so. By opposition, the *quality* of a member is her capacity to help people she certifies to respect the distance rule, independently of her level of involvement.
A member with a quality greater than or equal to *xpercent* (= 80%), makes, only by herself, people she certifiies respect the distance rule.

Tools

Parameters: Display the basic parameters of money.

Revoked Identities: Display revoked identities with their aliases, public keys and hashes, and the dates of their entries.

Missing Memberships: Display identities whose memberships were'nt renewed, with their aliases, public keys and hashes, the dates of their entries and their limit dates before revocation.

Identities: Display identities of all members with their aliases, public keys and

hashes, the dates of their entries and their limits of validity.

Newcomers: Display identities of all newcomers with their aliases, public keys and hashes, and the limits of validity of their applications.

Certifications From...: Display all certifications in the blockchain, sorted by senders, with their inscription dates.

Certifications To...: Display all certifications in the blockchain, sorted by receivers, with their inscription dates.

Sentries: Display the identities of sentries.

Distances: Give the distances of all members to the web of trust (proportion of sentries reachable in at most *stepMax* (5) steps) sorted by proportions (with drawing) and by aliases.

Qualities: Give the qualities of all members sorted by qualities (with drawing) and by aliases.

Centralities: Give the centrality levels of all members sorted by centrality levels (with drawing) and by aliases.

Limits of Memberships: Display the validity limits of memberships for all members, sorted by dates.

Limits of Missing Memberships: Display the limit dates before revocation of identities with not-renewed memberships, sorted by dates.

Limits of Certifications: Display the validity limits, for lack of certifications, of memberships for all members, sorted by dates.

History

Number of Members: Give the list of block dates where the number of members changed and the corresponding members' number since the beginning of the money (with drawing).

Flux of Members: Changes in the number of members by time unit (i.e.: one month). So, it's the derivative of the number n of members with respect to time t : $\frac{\mathrm{d}n}{\mathrm{d}t}$ (with drawing).

Flux of Members per Member: Flux of members divided by the number n of

members: $\frac{\mathrm{d}n}{n\,\mathrm{d}t} = \frac{\mathrm{d}\ln(n)}{\mathrm{d}t}$ (unit: percent by time unit). With drawing.

First Entries: Give the list of block dates where the number of members entering the WoT for the first time changed and the corresponding number *e* of first entries since the beginning of the money (with drawing).

Flux of First Entries: Changes in the number of first entries by time unit (i.e.: one month). So, it's the derivative of the number *e* of first entries with respect to time t : $\frac{\mathrm{d}e}{\mathrm{d}t}$ (with drawing).

Flux of First Entries per Member: Flux of first entries divided by the number *n* of members: $\frac{\mathrm{d}e}{n\,\mathrm{d}t}$ (unit: percent by time unit). With drawing.

Losses: Give the list of block dates where the number of members changed and the corresponding differences $l = e - n$ between the number *e* of first entries and the number *n* of members since the beginning of the money (with drawing).

Flux of Losses: Changes in the losses by time unit (i.e.: one month). So, it's the derivative of the losses *l* with respect to time t : $\frac{\mathrm{d}l}{\mathrm{d}t}$ (with drawing).

Flux of Losses per Member: Flux of losses divided by the number *n* of members: $\frac{\mathrm{d}l}{n\,\mathrm{d}t}$ (unit: percent by time unit). With drawing.

Use it and enjoy! - ¡Úsalos y disfrútalos! - Bonne utilisation - Приятного использованияп - Powodzenia - Viel Spaß

Gérard Meunier