Wikto – how does it work and how do I use it?

Introduction

This document describes how to use Wikto to quickly and easily perform web server assessments. Before we start we need to know what Wikto does and what it does not do. Wikto is not a web application scanner. It is totally unaware of the application (if any) that's running on the web site. So – Wikto will not look for SQL injection problems, authorization problems etc. on a web site. It is also not a network level scanner – so it won't try to find open ports, or see if the web site is properly firewalled. Wikto rather operates between these two levels – it tries to, for instance, find interesting directories and files on the web site, it looks for sample scripts that can be abused or finds known vulnerabilities in the web server implementation itself. Oh – and Wikto is not just Nikto for Windows. The Nikto scan is only of its many functions (and it does the Nikto scans totally different than Nikto does).

For the discussion of this article we are going to look at Wikto version 1.61. This is the latest version of Wikto. The homepage for Wikto is on the SensePost research site at http://www.sensepost.com/research/wikto. On this page you'll find documentation, source code, the install shield … and the latest version.

Installation

To use Wikto to its full capacity you need to install the following:

- WinHTTrack (www.httrack.com) – a web mirroring tool (you can use the default install)
- HTTprint (www.net-square.com) – a web server fingerprinting tool (by default, Wikto looks for this in the *c:\tools* directory, but you can configure it).
- And of course Wikto itself…

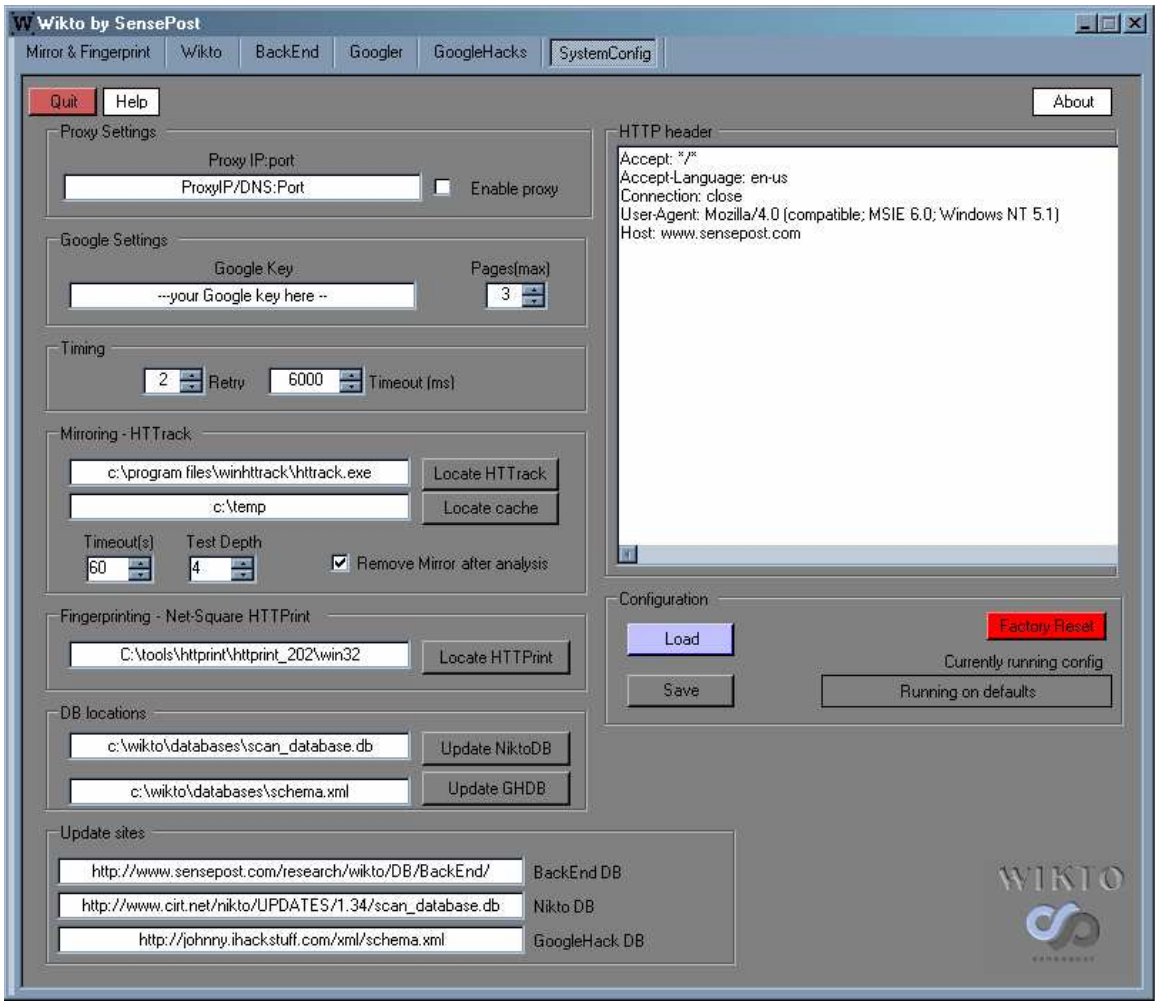
Design principals in Wikto

Wikto was coded in .NET C#. It has a graphical user interface. It uses text boxes to store results – these text boxes are "hot". It means that, should you wish to change any entry in a box, you could do so – and the interface will immediately assume the change. Results are never stored in memory. Wikto makes use of tabs for different section of the tool. At any time you can switch between tabs and use different functions of Wikto – at the same time.

Let's get into the meat of Wikto by looking at the different tabs:

System Configuration

Before you can use Wiko you need to configure it – it seems like lots of work but its not! When you run Wikto for the first time click on the SystemConfig tab - the configuration screen should look like this:

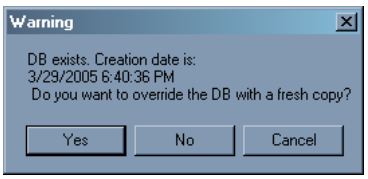


At first, this screen looks overwhelming - but relax – it's really not that bad. Every setting within Wikto is controlled from this screen – so it's really important that we configure it properly before starting a scan. Let's go through each section.

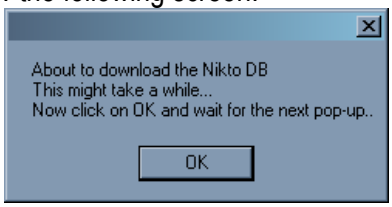
- **Proxy settings:** If you are located on an internal network that only allows you to connect to the Internet via a proxy you should enter the proxy's IP number followed by a semicolon (:) and the port. You can quickly switch between proxy and non-proxy mode by checking the "Enable proxy" checkbox. Remember – this checkbox is "hot" – if you change it during a scan Wikto will immediately implement the change.
- **Google key:** For the Google Hack section and the Googler (more on that later) Wikto uses the Google API. This API is implemented as a web service at Google, and Google only allows a certain amount of queries per person per day. Google enforces this by making use of a key that needs to be passed with every query. To use the API you need to register

with Google – after registration they will email you your Google API key. The key is good for a 1000 queries per day. You can register for your key at http://api.google.com

- **Timing:** this sets the timeout on read and write of connections made – note – it is not the connection timeout! The retry count sets how many times Wikto will retry a particular connection before giving up.
- **Mirroring – HTTrack:** Wikto makes use of WinHTTrack (http://www.httrack.com) to perform web mirrors (later more on that). This text field set the location of the executable. You may click on "locate HTTrack" to find it manually. The cache directory is used as a temporary storage space of web mirrors – set this to any directory where there's enough space. The timeout here is used during the mirroring process – in most cases you don't want to mirror the ENTIRE site. After the selected number of seconds the mirroring process will stop – on slow links this value should be increased. The test depth set how many link levels the mirroring process must follow. The mirroring process obviously stays on the site itself – it ignores links to other sites.
- **Fingerprinting – Net Square HTTPrint:** Wikto uses Net Square (http://www.net-square.com) 's HTTPrint for doing fingerprinting of web servers (later more on that). The directory in the text field should contain both the httprint executable (httprint.exe) and the signature file (signature.txt). You may use the "Locate HTTPrint" button to locate the correct directory.
- **DB locations & Update sites:** These two sections work together. The DB locations tell Wikto where to locally store the Nikto scan database and the GHDB (Google Hack DataBase). The Update sites text boxes hold the location of these databases on the Internet – so if the URL of these databases should ever change you may update it in Wikto. When pressing the "Update NiktoDB" or "Update GHDB" buttons the following will happen. Wikto looks at the URL for the relevant DB, connects to the site, and attempts to download the content to the file location. Before overriding the file it will prompt you with the time of the last update:



If you select "Yes" it will show the following screen:



After clicking on OK it appears as if nothing happens – Wikto is in fact now connecting to the URL and trying to download the database. When it completes the download the following screen appears:

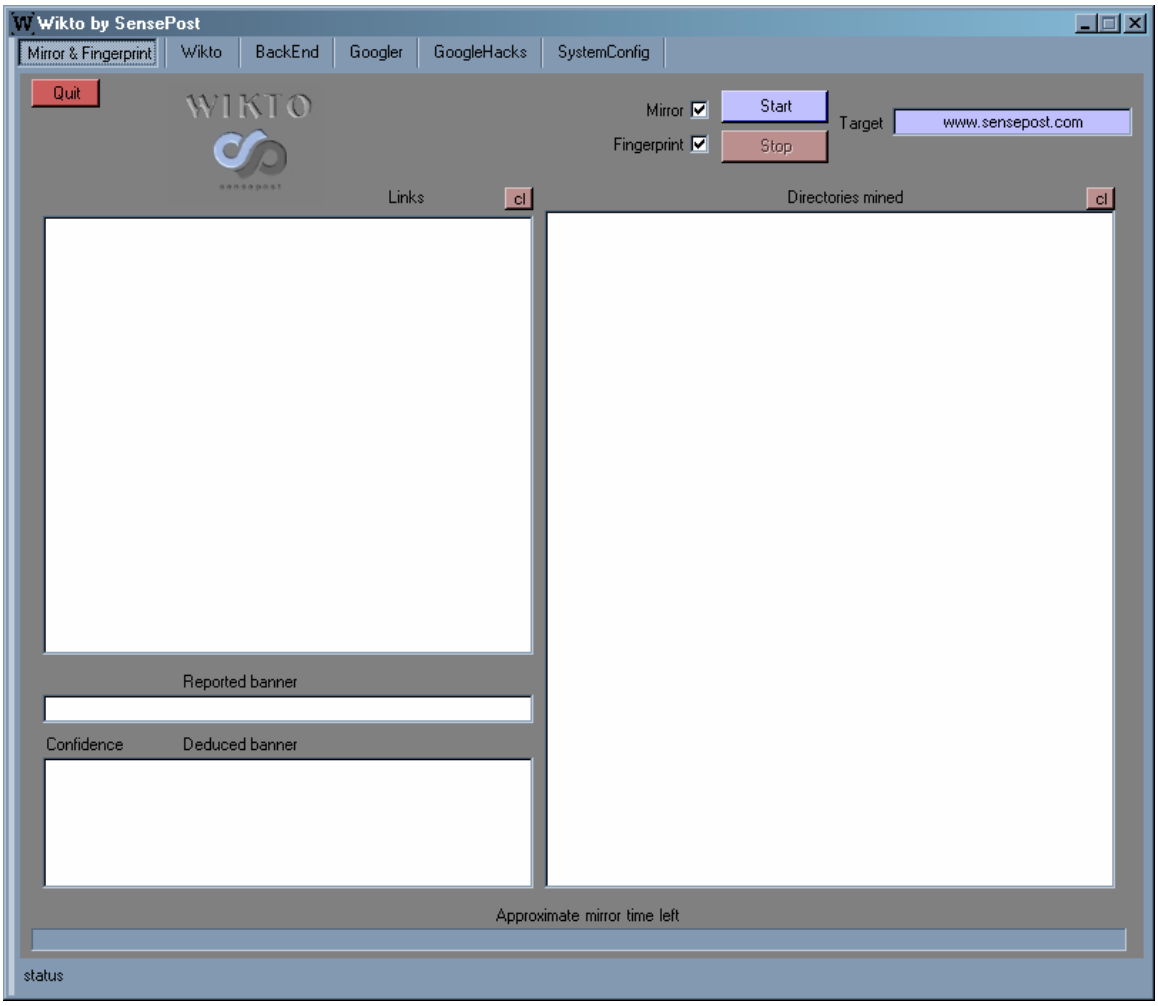
If you have changed the location of the local database, you should save the configuration at this stage (more on saving configurations later). The default values for this should be fine. If, for some reason (e.g. your provider blocks the site) you cannot get the database automatically you can always simply manually replace the files at their relevant locations.

- **HTTP header:** This is the HTTP header that will be used for every request. Note that the content length is not set (for POSTs) as this is calculated automatically. The use of this field is very important, and you can get quite creative with it. If the site you are connecting to for instance requires basic authentication, you can insert the credentials (base64 encoded of course) into the header. If you are working on a specific virtually hosted site you can populate the "Host:" line – thereby redirecting Wikto to the correct virtually hosted site. In normal operation Wikto automatically sets the header correct – so leave this as default unless you know what you are doing.
- **Configuration:** After you have configured everything you are now ready to save your configuration. Click on the "Save" button and in the file dialog (although it says "Open"…I know this confusing) select a appropriate name and click on "Open" (yeah yeah yeah…I know…confusing – we'll fix it in a next release). Your configuration is now saved and you can get to that configuration at any time by clicking on the "Load" button. Wikto however saves the location of the current configuration in the registry, and upon startup will ready the most recent configuration file.

With the configuration now in place we can start scanning! The first step would be to go the "Mirror & Fingerprint section":

Mirror & Fingerprint

Opening this tab leads to the following screen:



How does it work?

Before we start – why do we want to mirror the site at all? And what's with this fingerprint?

- We mirror the site because by inspecting links internal to the site we can find directories on the server. For instance – if the front page of the site has a link called "Login here" that points to http://site/cgi-bin/login/login.pl, we know that:
    - The /cgi-bin directory must exists
    - The /cgi-bin/login/ directory must exists
    - The server uses PERL scripts (the .pl extension shows this)
- By inspecting links externally on the site we might learn more about the site and/or the network itself – especially when some link deep inside the site links to http://10.0.0.1/intranet/welcome.asp !

- We fingerprint the site in order to determine the real web server type (and we determine the web server type in order to focus our attacks on the particular server type). Some administrators try to hide their true server type by changing its banners. Using sophisticated techniques, the good people at Net-Square has found a way to almost always determine the true identity of the server type. The techniques that they use is

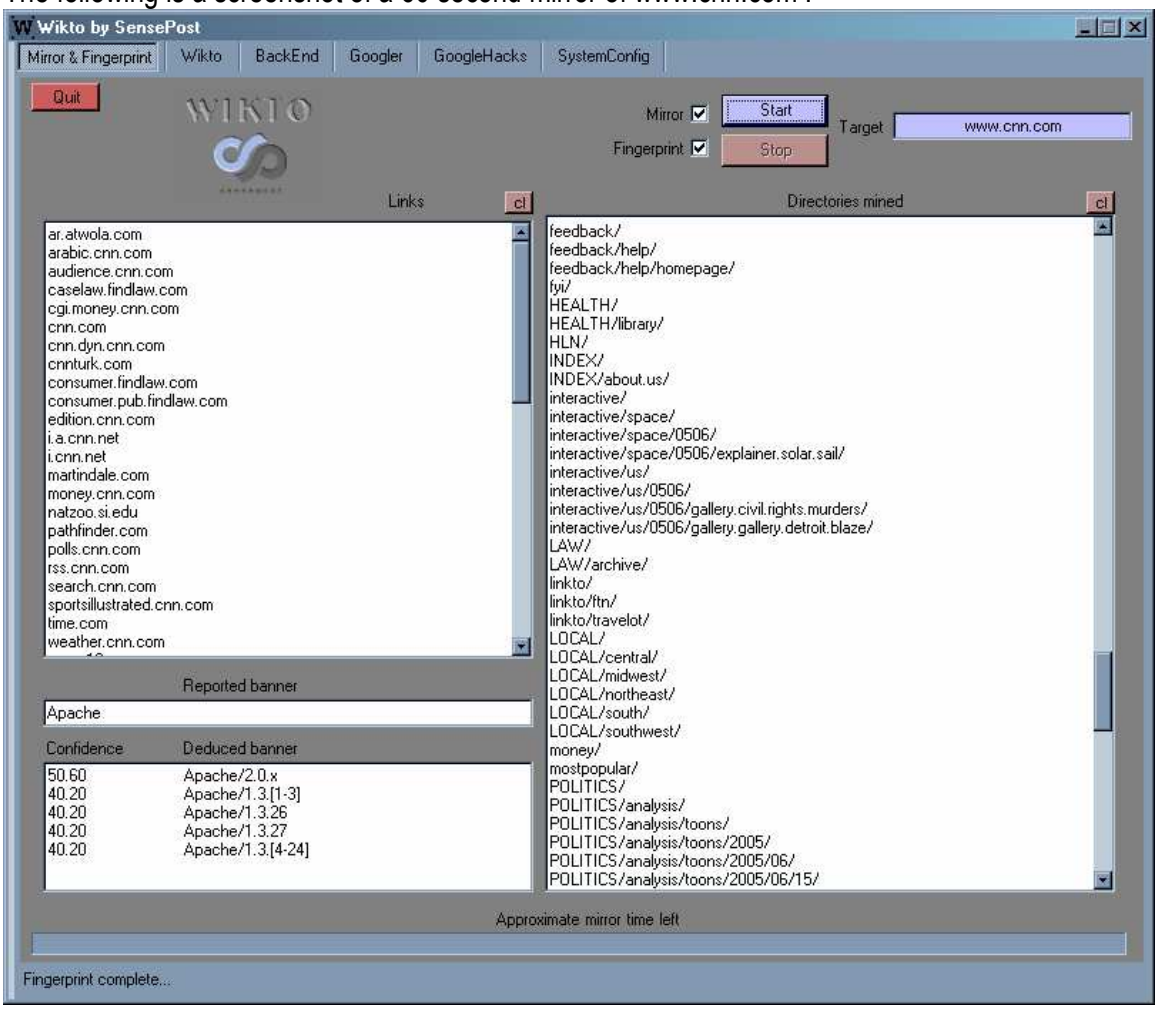
beyond the scope of this document, but is a really good read – you can find more on their techniques on their web site – http://www.net-square.com.

A very practical example where the use of the mirror greatly assists the scanning process is where the entire site is located underneath a single directory. In others words – the front page of the site http://www.xyzbank.com redirects to http://www.xyzbank.com/xyz/, and the rest of the website resides under the /xyz directory. If we don't know about the /xyz directory we wouldn't know to look for /xyz/admin when looking for administrative backend (more on that later).

How do I use it?

Step 1: Make sure you have installed WinHTTrack.
Step 2: Make sure your configuration is correct
Step 3: Enter the web site's DNS name or IP address in the Target text box
Step 4: Press start
Step 5: Sit back, relax and have some tea

The following is a screenshot of a 60 second mirror of www.cnn.com :

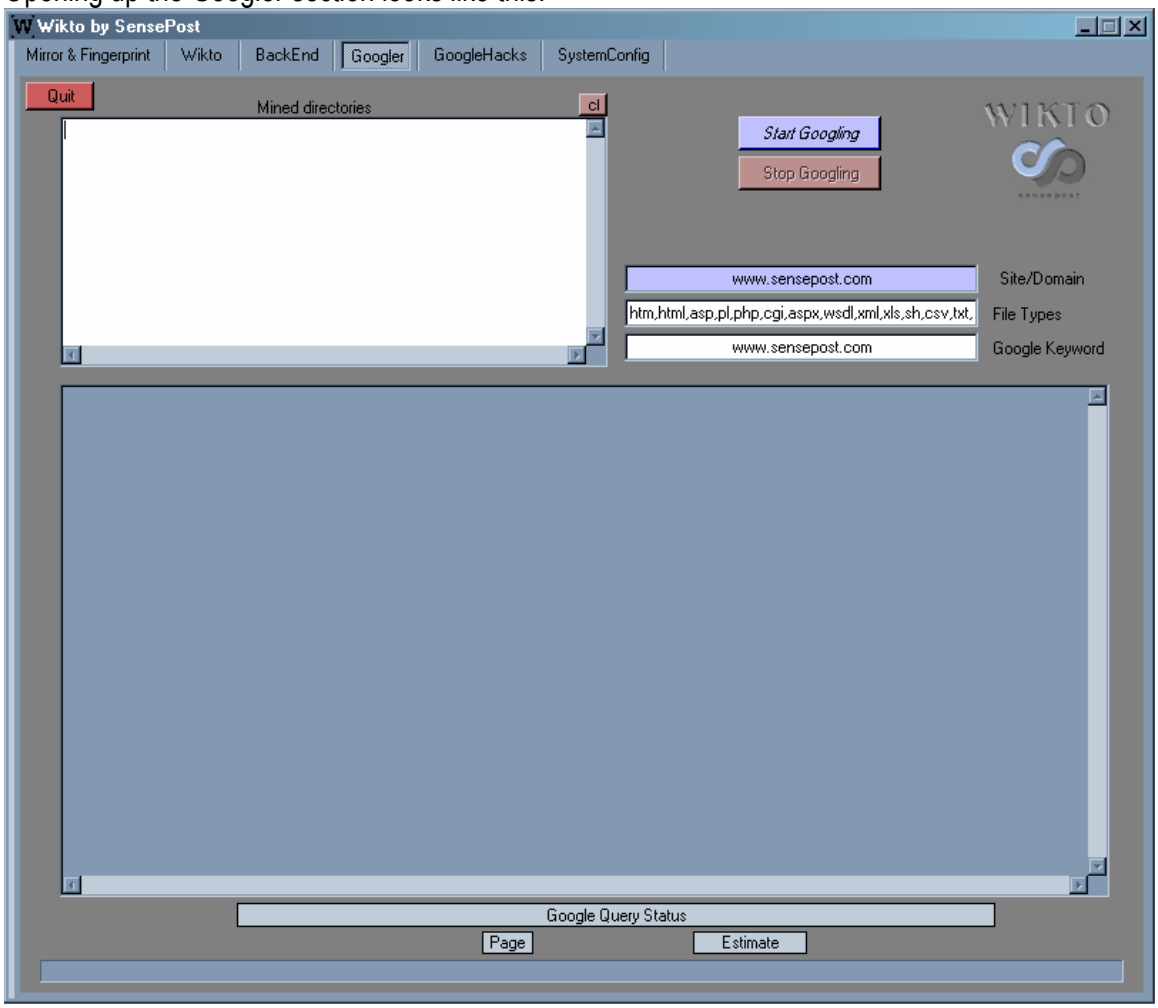


In this screenshot you can see that a great deal of directories were extracted. These directories are very valuable in later stages of the scan. You can also see that, although CNN reports its banner simply as “Apache” it is most likely part of the Apache 2.0 family. It is further interesting to see all the other CNN related sites that is linked from the main site – this information is especially valuable in a foot printing exercise.

Jumping around a bit we are next looking at the Googler section:

Googler

Opening up the Googler section looks like this:



How does it work?

The Googler performs two main functions:

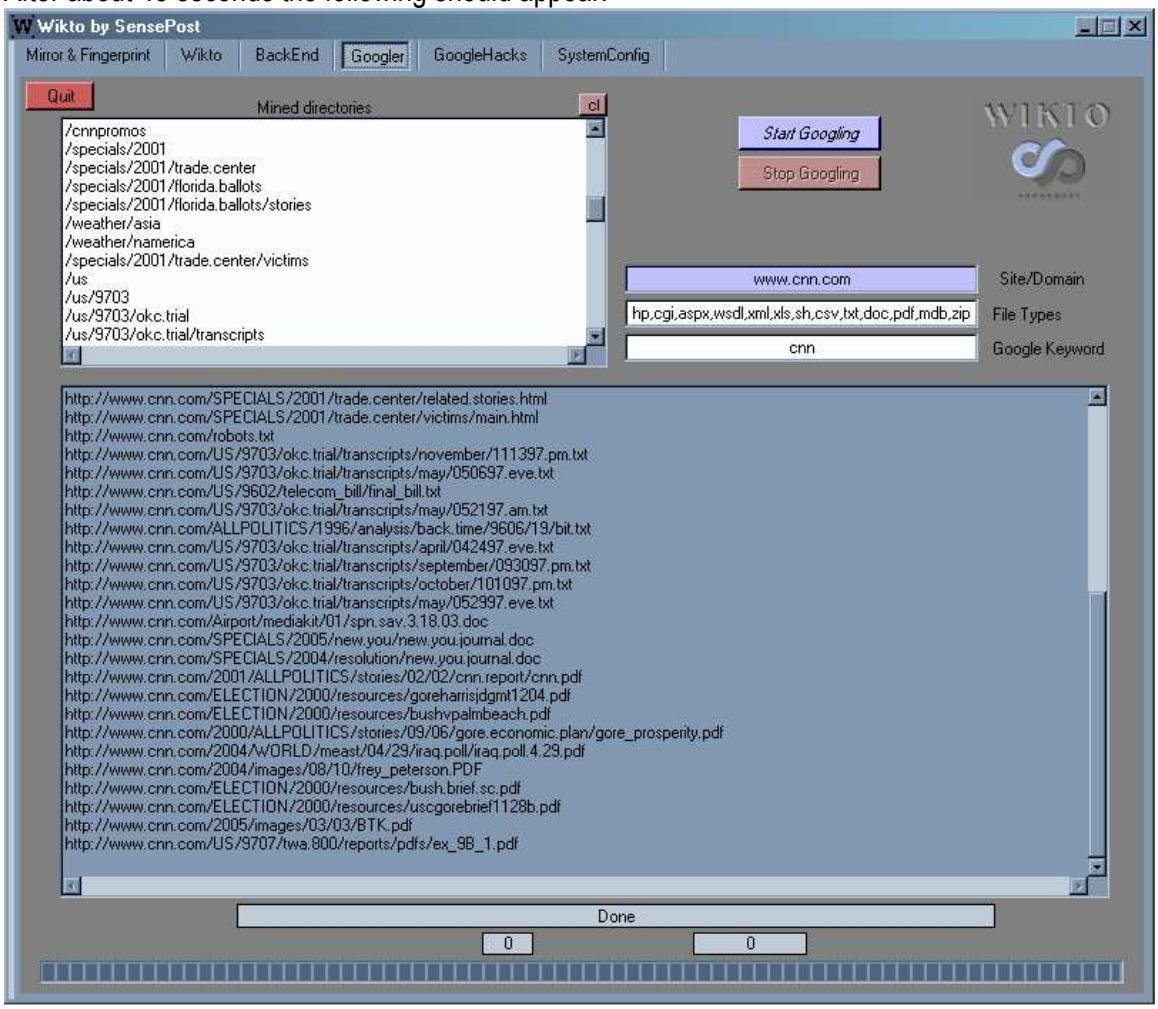
- Just like the Mirroring tool it find directories on the site – it does this by basically Googling for a keyword (Google keyword) as well as a file type on a certain site (or domain, or subdomain) and extracting directory names from resulting URLs it found. So – its like the mirroring tool – just not as thorough…but a lot quicker – and with a lot less noise on the target site (as it never touches the site itself)

- It looks for juicy files on the site (or on an entire domain). By combining the "site" directive with the "filetype" directive we can ask Google to find us any MS Word files it ever indexed on a site…or ZIP files…or MDB files.

How do I use it?

Step 1: Make sure you have a valid Google key
Step 2: Make sure your configuration is correct – you may choose to make the Google test depth deeper – but the 80/20 rule applies here
Step 3: Enter the site in the target text box – the Google keyword will automatically assume the same value
Step 4: Edit the Google keyword. In the example we will use www.cnn.com as the target, and edit the keyword to be only "cnn".
Step 5: Press start
Step 6: Wait…

After about 45 seconds the following should appear:

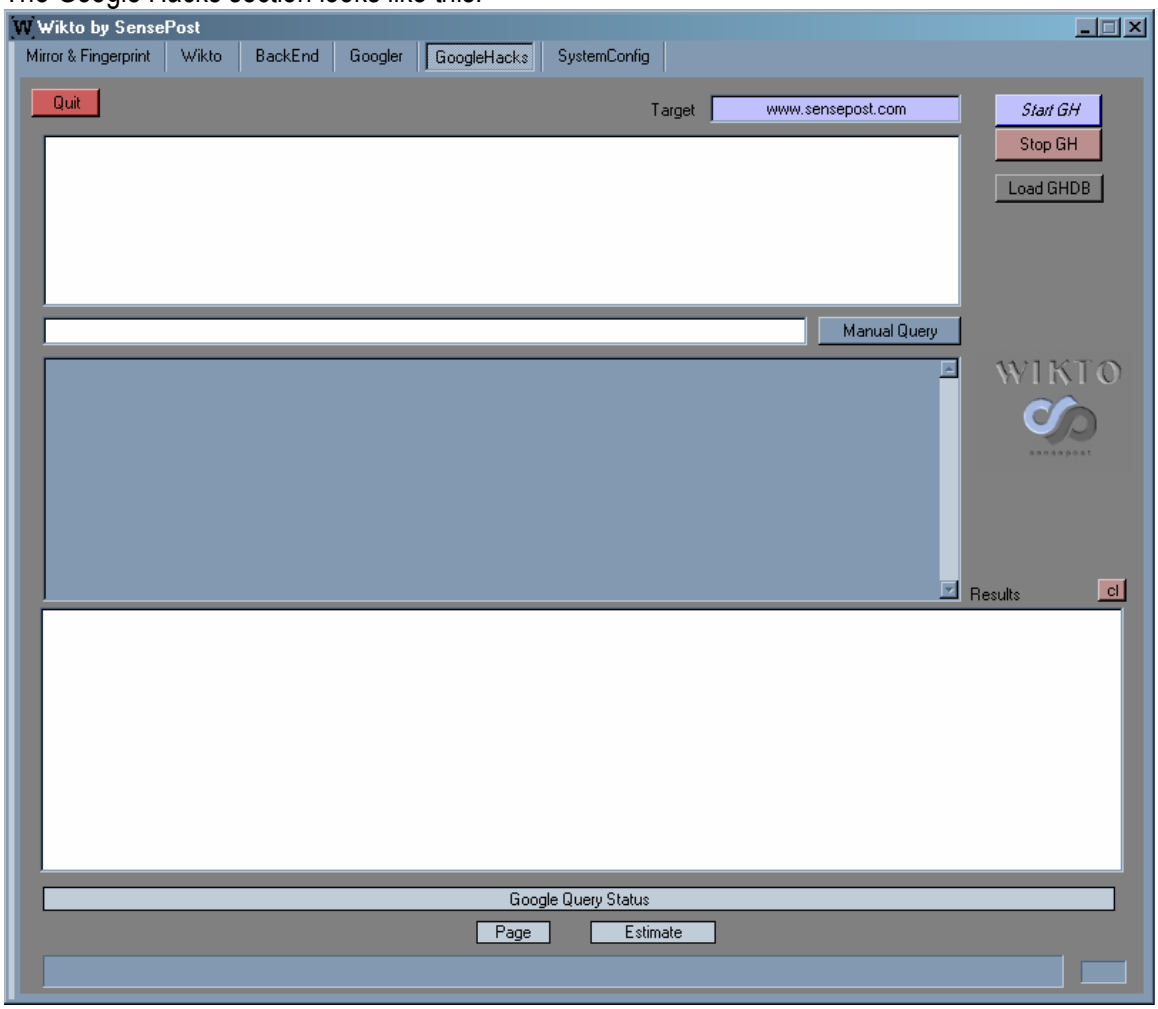


As with the Mirror we can see that a lot of directories were mined from the returned URLs. We can manually inspect the list of found resources for interesting files.

Staying with Google – let us look at the now famous Google Hack section:

GoogleHacks

The Google Hacks section looks like this:



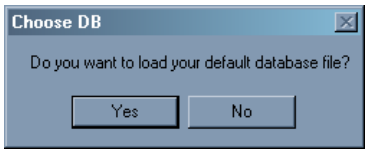
How does it work?

The whole Googling Hacking this is a very hot topic at the time of writing this document. What is it? Why is it interesting? Well – let's see… Google Hacking is no more than simply performing Google searches with interesting terms. Suppose you Google for the word "secret" – do you think you'll find secret a secret document? Most likely not. But what if you combine this with a "site" directive – thereby limiting results to a certain web site or domain? Still not very likely. And if you combine it with the "filetype" directive – limiting it to … let's say .DOC files? Now it becomes interesting. This is what Google Hacking is all about – it has evolved to the point where a Google Hack is now tuned to return entries in Google that match the default installation page of a web camera. Johnny Long is the king of Google Hacking and has even written a book called "Google Hacking for Penetration Testers" (check out his site at http://johnny.ihackstuff.com). Being a good friend of mine we spoke

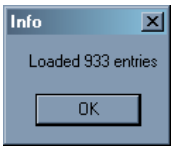
after BlackHat Vegas 2004 and decided that it would be fun to pull all these signatures into Wikto. By using the site directive and the Google API Wikto now automates the process while focusing it on a particular site or domain.
The GoogleHack section of Wikto thus performs all the different searches for us and provides us with the entries within the Google database.

How do I use it?

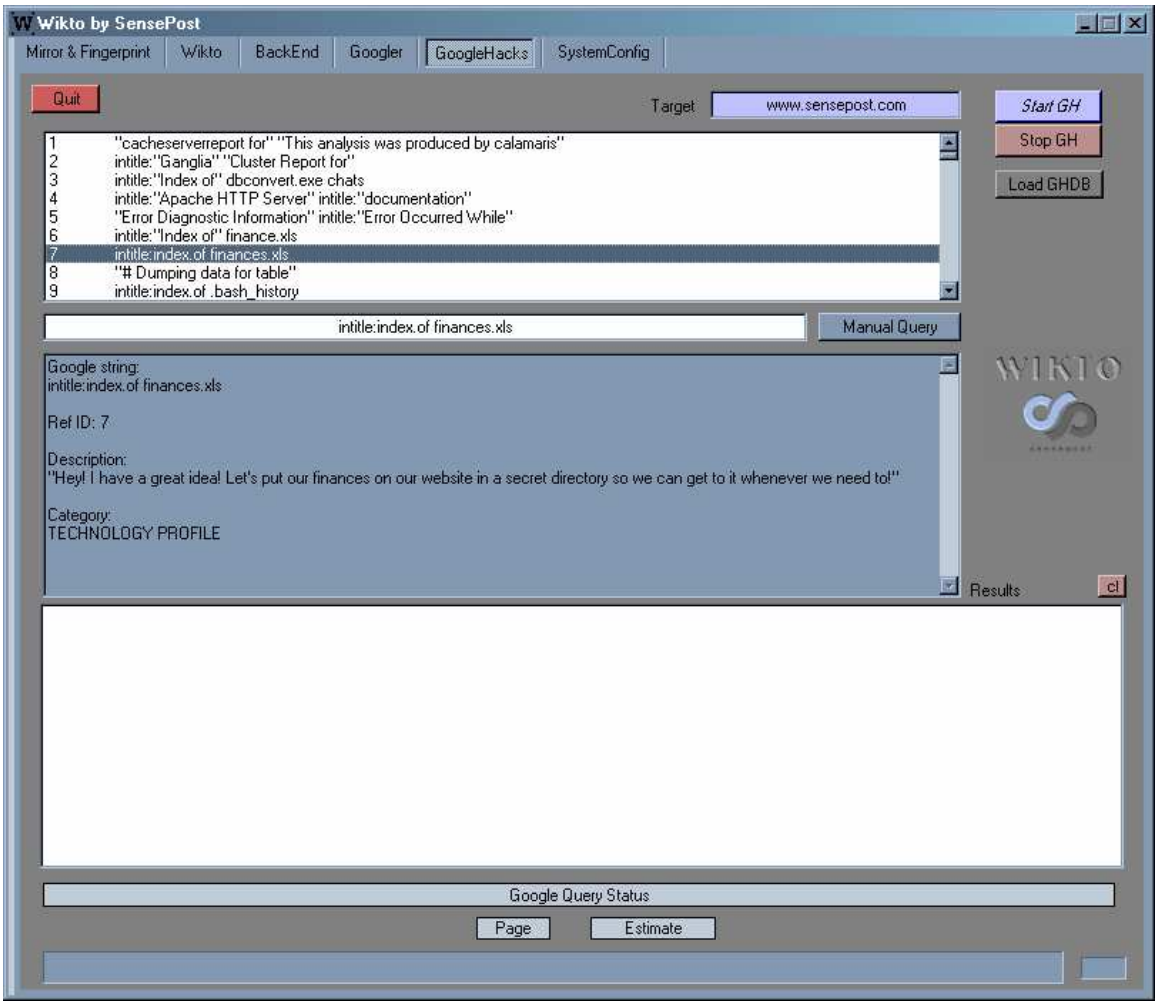
Step 1: Make sure you have a copy of the GHDB locally – see the System Config section
Step 2: Make sure your configuration is correct – especially the location of the GHDB.
Step 3: Click on the "load GHDB" button to load the Google Hack Database. You should see a screen that prompt as follows:



Wikto is asking you if it should load the database as defined in the System Config section. If you have another database that you would like to load you can click on "No". Most likely you will simply click on "Yes".
Step 4: After clicking on "yes" you should see



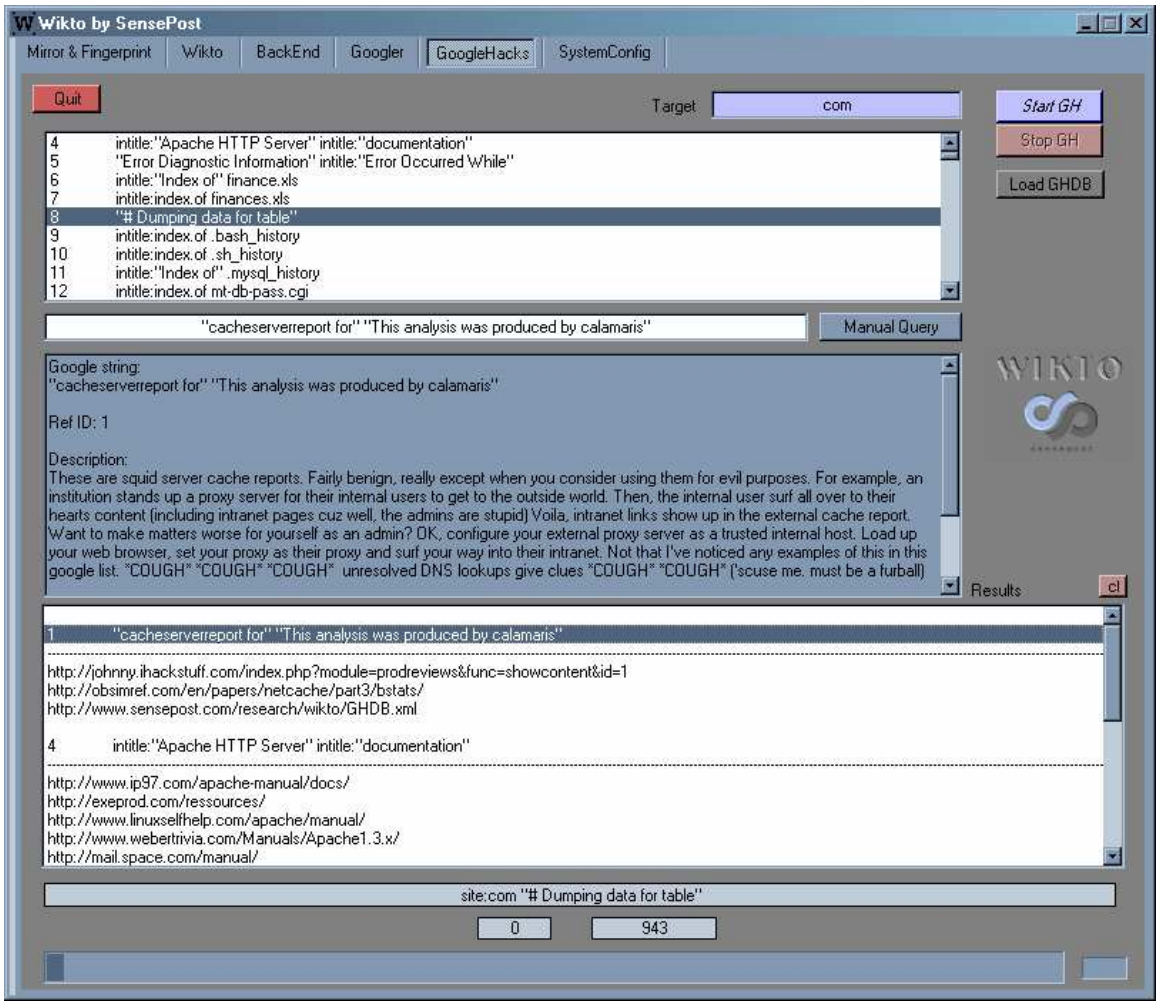
After clicking on "OK" the GHDB should be loaded in the application:



By clicking on any of the entries we can see a detailed description of the Google Hack in the centre text box. You may browser the Google Hacks at leisure and giggle at some of the entries.

Step 5: enter a target site or domain
Step 6: Click Start
Step 7: Wait..sit back and watch the show.

As Wikto works through the database it will show the search term in the status bar at the bottom. It will also highlight the entry its working on if your mouse is located inside the top listbox. As Wikto finds hits on the Google Hacks it will display them in the lower list box. The Google Hack entry and the resultant URLs are displayed:



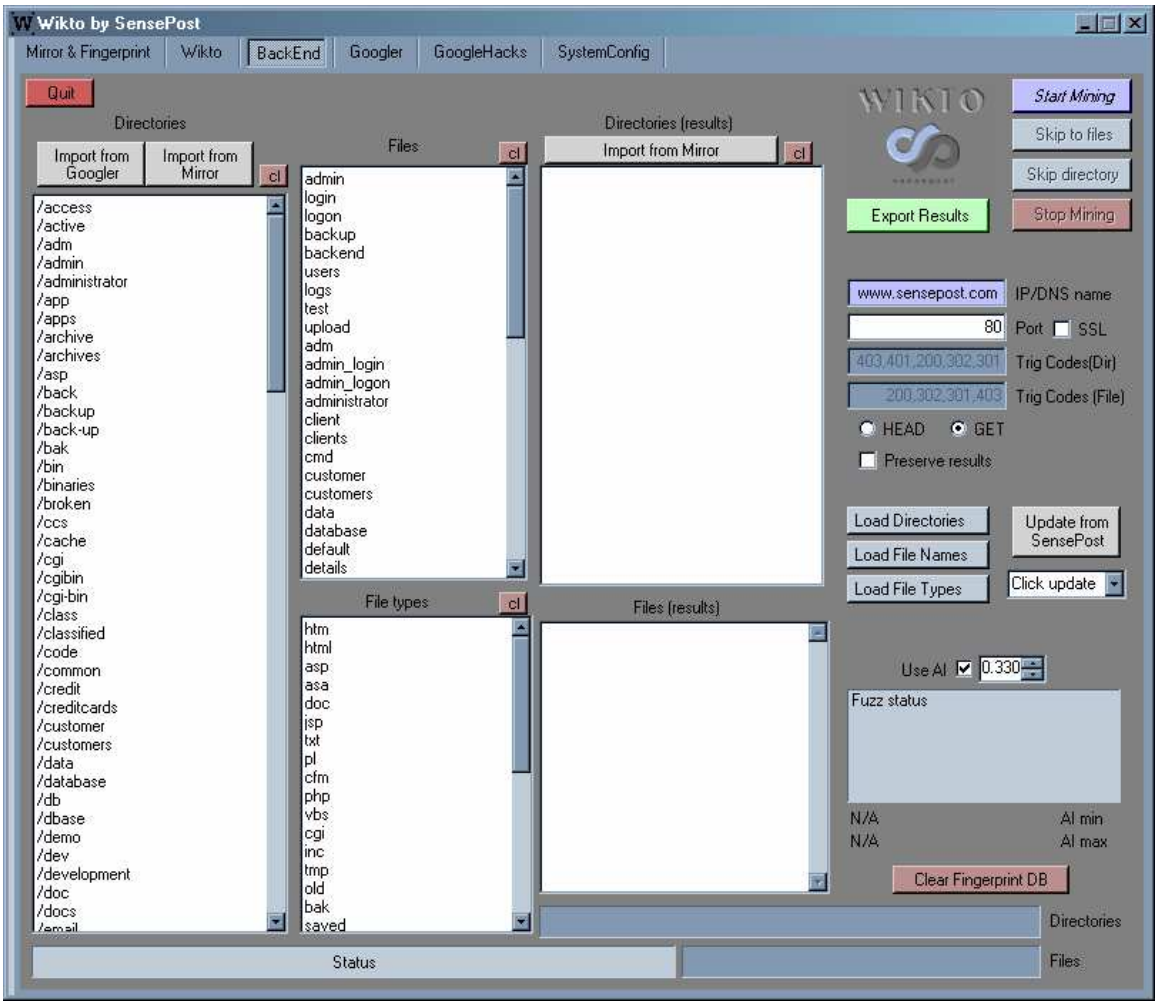
The description text box will always hold the description for the most recently chosen Google Hack – be that from the database, or the results.

As a simple extra a manual query text box was added – this allows you to literally search for anything and see only the resultant URLs coming back. This works well for finding your own Google Hack!

With these sections under control – let us have a look at the more interesting tabs…

BackEnd

When the BackEnd tab is opened it looks like this:



How does it work?

The BackEnd is actually very simple – it tries to find interesting files and directories on a web server. The thinking is that a web server might have the /secret directory, but that it is not linked from any page on the server (else it would have shown up on our mirror right?). The same goes for administrative backend interfaces (therefore the name BackEnd). This concept can be further extended to files – if we have a list of directories on the web server we can start looking for interesting files in these directories.

The Backend Miner does two things relatively smart:

- It looks recursively through directories – this means if it found the directory /source, it would check /source for the existence of all the other directories within it – e.g. /source/admin.
- It uses fuzzy logic to determine if a file/directory is present. Without going into the nasty details of the algorithm, it basically does the following:
    - It requests a file/directory that definitely does not exist – e.g /hierdieisnooithiernierig
    - It stores the reply from the server

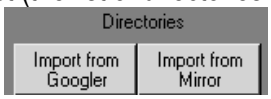
- It requests the real file/directory – e.g. /cgi-bin
- It now compares the two responses – if the response is the same the index will be high- this would mean that the file or directory does not exist. If the response is different the index will be low – this would mean the file/directory exists.
- Wikto now checks if the index is less than the trigger level – if so the directory or file is reported.

How do I use it?

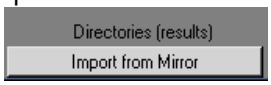
Step 1: Configure the directories/files that you want to check.
Wikto comes standard with a set of directories and files and extensions that it would check. These are stored in normal text boxes – so feel free to add or delete entries as you wish. As part of Wikto SensePost provides a couple of categories of directories/files/extensions which can be downloaded internal to the tool. To see a list of categories click on the "Update from SensePost" button once. You should see the following:

Choose a category (currently we have Standard, Quick and Full) and click on the "Update from SensePost" button again. You should see that the directories, files and extensions are updated.

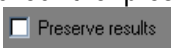
The Mirror and Googler tool now comes into play. You have one of two options – you can either import the directories into the check list (the list of directories on the left)



, or (in the case of the Mirror import) import it into the results:

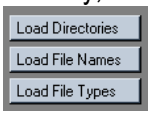


If you import it into the results you should check the "preserve results" checkbox

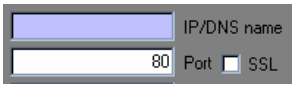


so that Wikto does not clear the list before it starts. Bottom line – if you are in a hurry and you just want to check for interesting files in known directories then you can import the directories in the result text box and when Wikto starts click on the "Skip to files" to go straight to the file checking. If not – do the proper thing and import both in the left hand text box.

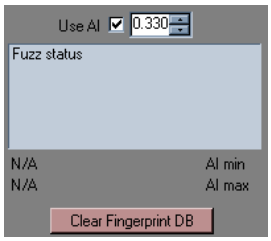
If you have a special list of directories, files or extensions you want to check you can either copy and paste it into the relevant text boxes … or finally, use the "load" buttons:



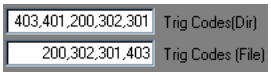
Step 2: Enter the target, the destination port and if the site is SSL enabled. For an normal SSL site you should enter the port as 443.



Step 3: Decide how the triggers should work. You can either use fuzzy logic (if you are seeing a lot of false positives) or you can go with the standard status codes. If you are using AI/Fuzzy logic you need to set the threshold (as explained in the previous section). As Wikto goes along it will show you the minimum and maximum index values, as well as the "Fuzz Status" – this is basically telling you if the a response fingerprint is needed or not. At any time you can clear the fingerprint DB – this will remove any stored responses. If you are starting a scan on a new site you obviously need to clear the DB!



The more paranoid you are about false positives – the lower you should set the value…but if you set it too low you wont get any results. Trail and error…
You may also choose not to use AI/Fuzzy logic – if the site is reporting clean status codes (e.g. no "friendly 404s") there is really no need to use the AI settings. If you want to, you can set the status trigger codes but the defaults should work fine.



Step 4: Click on the "Start Mining" button. Wikto will now start with directory mining and will do so recursively (see previous section). When no more directories can be found it will start working on files – it will check per found directory, per filename and per extension. If there are many directories this process can take several hours…

Because of this Wiko gives you the ability to skip certain directories or..skip directory checking altogether and move straight into file checking. Even when it is checking files within a specific directory for files you can tell it to skip the directory and move on to the next one. To advance one directory click on the "Skip directory" button. To go directly to file checking click on the "Skip to files" button.

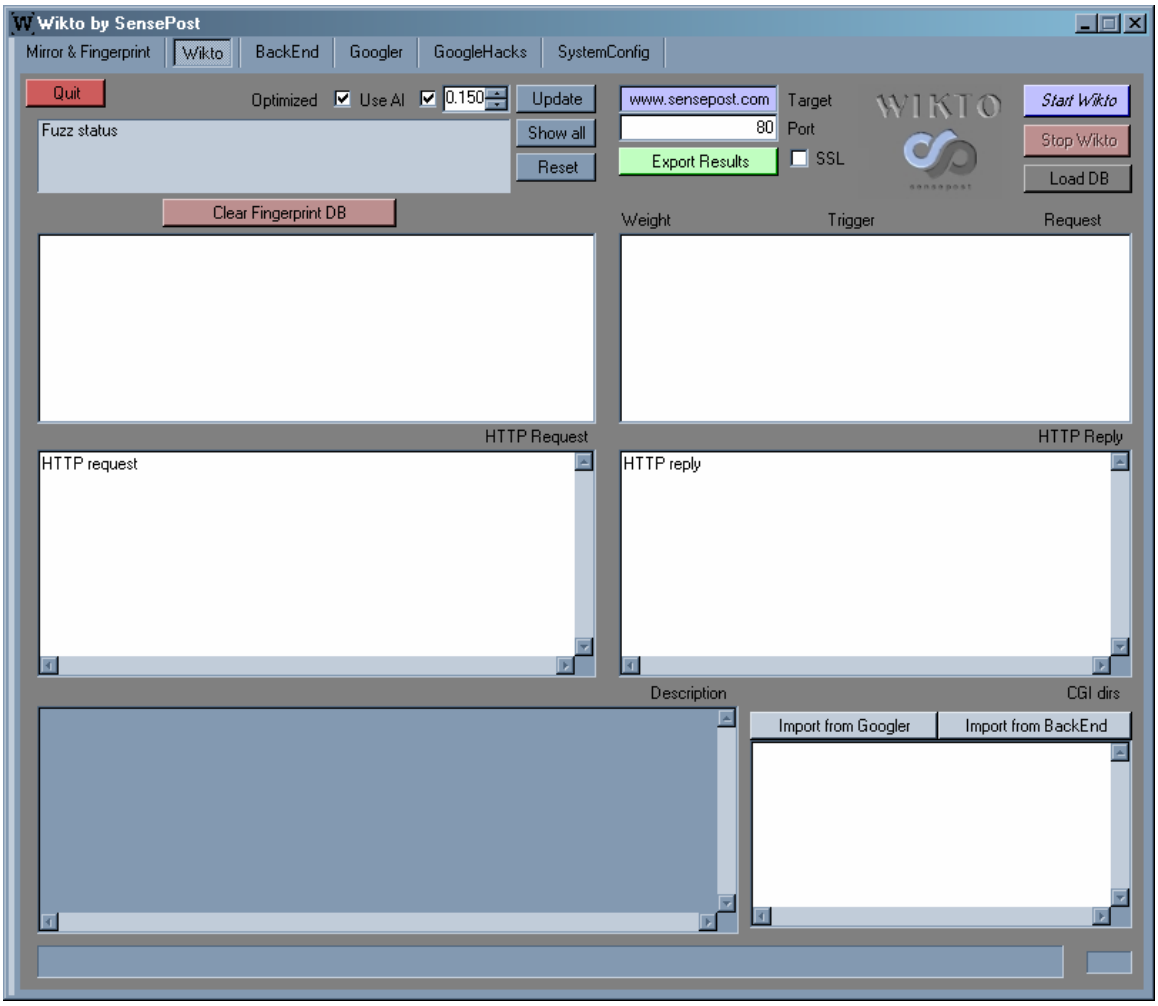
Step 5: wait..get lunch...

The results should be coming into the directory results text box..if an entry in the directory list has a blue foreground it means that the directory is indexable. When the directory mining is complete results should start getting into the file box.

Finally – we will look at the Wikto component of Wikto (how strange does that sound)

Wikto

The Wikto part looks like this:



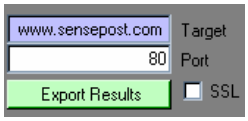
How does it work?

To understand the Wikto component of Wikto you need to understand what Nikto is. Nikto is a text based web server vulnerability scanner written in PERL by the good guys at CIRT – http://www.cirt.net. Nikto scans for over 3000 potential problems on a web server. It is beyond the scope of the document to explain what types of checks it performs – you can read all about it at http://www.cirt.net/code/nikto.shtml.

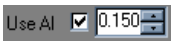
Like the BackEnd miner Wikto has the ability to perform fuzzy logic on the responses thereby greatly reducing the occurrence of false positives. Wikto also has the ability to import directories from both the BackEnd miner and the Googler.

How do I use it?

Step 1: Make sure you have the lastest Nikto database – if not download it (see SystemConfig section)
Step 2: Click on the "Load DB" button – as with the Google Hacks it will prompt you if you want to load the default database.
Step 3: Enter a target (IP or DNS name), a port and check the SSL checkbox if needed.

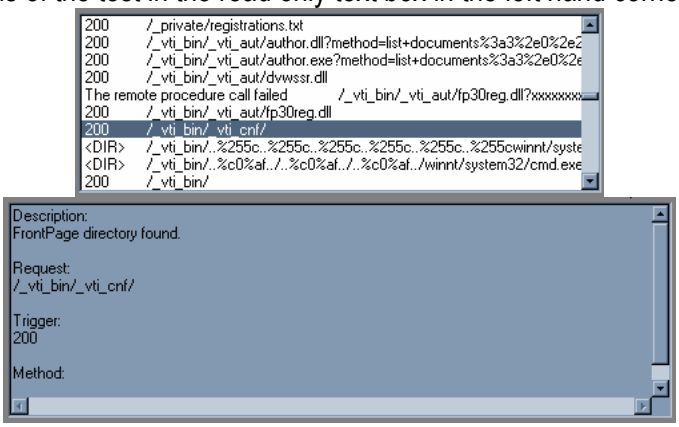


Step 4: Decide if you want to use Fuzzy logic/AI – you might want to test if it is indeed needed first. If you you find lots of false positives you need to switch AI on. In this case you'll need to set the trigger level (see the BackEnd Miner section for more information)

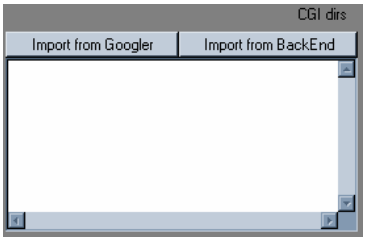


Step 5: Decide if the scan should be optimized. If the checkbox is checked Wikto will not perform tests when the test contains a directory that was not found in the BackEnd miner's results. For example, if testing for the presence of /cgi-bin/sample.pl and the test is optimized Wikto will check if the /cgi-bin directory was found in the BackEnd miner, and if not, it will skip the test.

Step 6: Before starting the test can click on any Nikto test (in the listbox) – when selected you should see the details of the test in the read only text box in the left hand corner



Step 7: In many of the tests you would see that the location of the test contains the letters "@CGIDIRS". This is a special symbol within Nikto that means "replace this with all the CGI directories found". Wikto gets these directories from the "CGI dir" text box in the right hand corner:



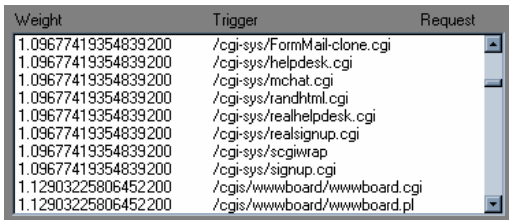
If this text box is not populated…Wikto will simple skip these tests. It is thus really important that you populate it. You can populate this text box automatically from the BackEnd miner or the Googler. Keep in mind that not all directories will be marked to be executable – but if you don't mind that it will take a little longer it wouldn't hurt importing all directories.

Step 8: Click on the Start button.

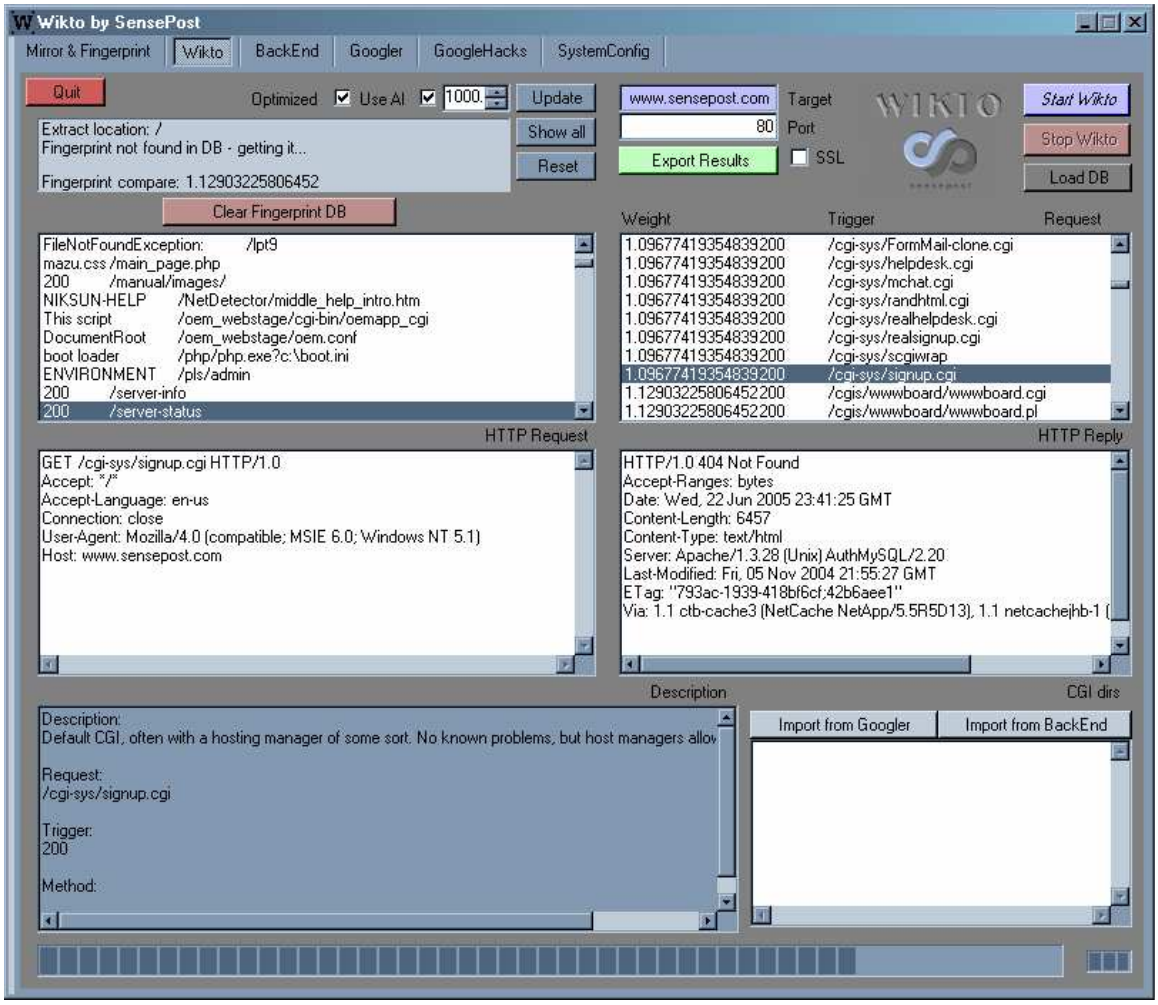
Step 9: If your mouse pointer is within the boundaries of the test list box you will see Wikto's progress as it goes through the tests. You will also see the actual HTTP request and the response – this is great for manual inspection of results.

Step 10: Tune the AI/Fuzzy logic trigger. If you are using AI/Fuzzy logic triggers Wikto will keep track of the indexes of each test. This means that you can adjust the trigger level in real time and see which tests resulted in an index below the threshold. To see this in action adjust the trigger level and press the "Update" button:

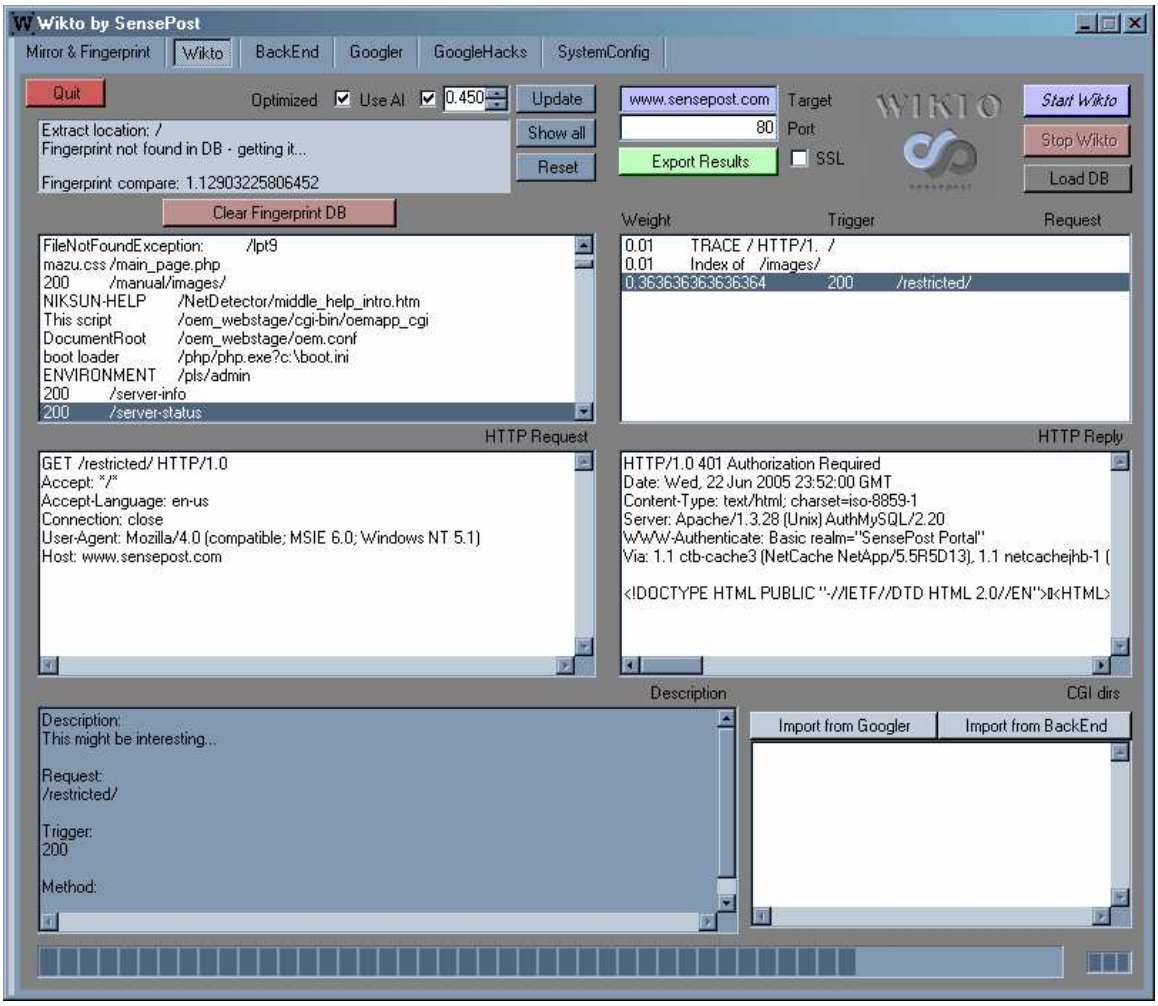
The “Show all” button sets the trigger level to 1000 - and no index will ever be above 1000. In practical terms that means that it shows all results:



If you click on any of the results you will see that the actual HTTP request and response is shown:



To reset the trigger levels to the previous level simply click on the “Reset” button. This will restore the trigger level and automatically perform an update of results:



In the screen shot above we see that the SensePost web site has the /restricted directory, but that it is protected by Basic Authentication (the Nikto scan database also contains a couple of checks for interesting directories).