**Virtual Host (vhost)**

**Last updated: February 22, 2017**

**What Does Virtual Host (vhost) Mean?**

**A virtual host is is a type of hosting service provider that focuses on virtual infrastructure solutions, including virtual servers, computers, storage and other hybrid platforms that enable the hosting of data, applications and/or services. It includes all technologies and service models that allow individuals and organizations to source computing infrastructure solutions and services from the Internet.**

**A virtual host also refers to a device that is remotely accessed by users for the hosting of data or software services.**

**Techopedia Explains Virtual Host (vhost)**

**A virtual hosting platform may be shared by different users or dedicated to a website, application or customer. In a shared virtual host platform, a provider’s physical server compute and storage resources are shared by two or more subscribers. Each host is provided with a predefined resource quota, which may be scaled according to provider and infrastructure capability.**

**What are virtual hosts?**

A web server can host more than one site. Each of this site is a virtual host. The name (identifier) of the virtual host can be:

* Domain name
* Subdomain (of any level, for example, kali.tools, en.kali.tools, test.en.kali.tools – these are all virtual hosts that can have different settings and even different IP addresses)
* Arbitrary names (any string that is not a valid domain name or subdomain)

Note: now I mean name-based virtual hosts, in addition IP-based virtual hosts and port-based virtual hosts exist.

About the search for virtual hosts, actually sites on one IP, as well as subdomains, I already wrote and I’ll give you links to these articles at the end of this instruction. Now I want to consider the option of searching for arbitrary host names, which is particularly relevant for local networks. But the tool I am talking about can also be used to search for domains and subdomains.

In a local network (for example, an organization’s corporate network), all connected computers can use the organization’s own DNS server. Such a DNS server normally processes requests for domains (that is, for example, it will send IP 185.26.122.50 to the miloserdov.org request), and can also be configured to process additional names.

For example, the system administrator can add there such names as

* **files.loc**– for local file sharing
* **forum** – for announcements and discussions
* **info.all** – web resource with information for all
* **info.man** – web resource with information for managers
* etc.

As a result, instead of the IP address, users of the local network can type **http://info.all** in address bar of a browser.

Or the following address: **http://forum/**

If you need to make these local hostnames work on the same computer, you do not need to use the DNS name server — you just need to edit the **/etc/hosts** file.

The peculiarity of such virtual hosts is that they can hardly be found using third-party services, since they can be purely local. Therefore, they have to look for brute force methods, and for this you need special tools, for example VHostScan.

**Installing VHostScan on Kali Linux**

Installing VHostScan is simple, but the official help for some reason indicates the use of Python3, which in practice makes it impossible to install some obsolete dependencies and the program does not start (I wasted a few hours trying to solve the associated problems). But when using Python2, everything runs smoothly:

|  |  |
| --- | --- |
| 1  2  3 | git clone https://github.com/codingo/VHostScan  cd VHostScan/  sudo python2 setup.py install |

**How to search for virtual hosts**

The essence of the search for virtual hosts is quite simple – a request is made to the web server with the indication of the hostname and the received response is analyzed. But there are a few pitfalls.

The VHostScan program is quite intelligent. It knows about the existence of default virtual hosts of the web server — these are the hosts whose contents are shown in case the requested host is not found. Thanks to this technique, the program does not show false results if the web server responds to all hosts in a row. And the program itself determines what exactly is shown by the default virtual host – based on the analysis of all the results obtained. By the way, for this reason it is useless to check one single hostname at a time — the program simply cannot find out if it really exists, or the default content is shown.

It happens that the web server when accessing non-existent hosts is configured to show similar, but not identical pages. An example of such a response:

|  |  |
| --- | --- |
| 1 | Host not found. The current time is 19 hours 18 minutes 30 seconds. |

Or

|  |  |
| --- | --- |
| 1 | Host\_NAME\_HOST not found |

Strictly speaking, such pages are unique and can confuse some programs, resulting in false results. In turn, VHostScan is able to work with this problem and analyzes the submitted pages in parts. That is, if the server sends something similar, the non-existent hosts will not be included in the list of found ones, or they will receive a low probability of existence.

**Search for virtual hosts by dictionary**

For the work of the program you need a target address and a dictionary. Moreover, several dictionaries are supplied with VHostScan and if you have not specified your own, the program will use the default dictionary. That is, only the target address is required. As an address, you can specify a domain, IP address or a range of IP addresses.

An example of searching for other virtual hosts with the default dictionary [virtual-host-scanning.txt](https://github.com/codingo/VHostScan/blob/master/VHostScan/wordlists/virtual-host-scanning.txt) on a web server running [hackware.ru](https://hackware.ru/):

|  |  |
| --- | --- |
| 1 | VHostScan -t hackware.ru |

Since this is a hosting service, no extra virtual hosts were found. By the way, the default dictionary is more suitable for the local network.

**How to brute-force for sites on a single IP**

In addition to searching for non-standard virtual host names, you can search for ordinary domains and subdomains. For example, create a file **vhosts.dic** and copy into it

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | mi-al.ru  mi2-al.ru  localhost  sdsdfsdf.ru  hackware.ru  www.hackware.ru  www2.hackware.ru  www3.hackware.ru  en.hackware.ru  test.hackware.ru  ssssdfsdfsd.org  suay.ru  one-more-fake.site  admin  mysql  local  suip.biz  www.suip.biz  test.suip.biz  test.en.suip.biz  forum  dash  administration |

To search for virtual hosts on the IP address ***(-t 185.26.122.50)*** with the dictionary ***(-w vhosts.dic)***:

|  |  |
| --- | --- |
| 1 | VHostScan -t 185.26.122.50 -w vhosts.dic |

To specify your dictionary, use the -w option. Results:

|  |  |
| --- | --- |
| 1  2  3  4  5 | [+] Most likely matches with a unique count of 1 or less:      [>] mi-al.ru      [>] hackware.ru      [>] test.hackware.ru      [>] suay.ru |

**Search for virtual hosts using SSL**

For a web server (using Apache for example), the address [http://suip.biz](http://suip.biz/) and [https://suip.biz](https://suip.biz/) (they differ only in the HTTP and HTTPS protocols) are two different virtual hosts! Of course, when setting up a web server in such cases, the same files are usually specified as the contents of the site, but you need to remember such an unusual opportunity – to show different sites with the same domain name, but with a different protocol (if more precisely, these are two port-based virtual hosts for the web server). It may be the case that for one of these protocols (including HTTP), the virtual host setup is simply not available.

To use the secure HTTPS protocol there is an option **--ssl**. By default, VHostScan uses port 80 and the option **--ssl** does not change this! That is, in addition to this option, you also need to specify the correct port number with the **-p** option:

|  |  |
| --- | --- |
| 1 | VHostScan -t 185.117.153.79 -w vhosts.dic --ssl -p 443 |

The result:

|  |  |
| --- | --- |
| 1  2 | [+] Most likely matches with a unique count of 1 or less:      [>] suip.biz |

**Hot to get hostnames from standard input**

You can pipe hostnames from other tools to VHostScan. The program will interpret this data as a list of words, for example:

|  |  |
| --- | --- |
| 1 | cat bank.htb | VHostScan -t 10.10.10.29 |

If you wish, you can combine the use of the dictionary with the standard input. In this case, the lines from the dictionary will be added to the lines piped from another program:

|  |  |
| --- | --- |
| 1 | echo -e 'a.example.com\r\nb.example.com' | VHostScan -t localhost -w ./wordlists/wordlist.txt |

Notice how the string is passed with **echo**:

* the **-e** option is used
* as the domain separator is used the string **\r\n**

## BaseHost Substitution

The program supports the insertion of the basehost on the fly. To do this, use the string **%s** in the dictionary, for example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | admin.%s  beta.%s  dev.%s  development.%s  m.%s  mobile.%s  old.%s  secure.%s  www.%s |

You can create a separate dictionary with such strings, or use it in conjunction with regular static strings. The **%s** will be replaced with the base host, which must be specified with the **-b** option. If you did not specify a base host, then what is specified as a target with the **-t**option will be substituted – this can be either a domain or an IP address.

## Port Forwarding

VHostScan is suitable for use with **pivoting** techniques as well. By the way, see the article ‘[Network pivoting: concept, examples, techniques, tools](https://miloserdov.org/?p=2973)’.

If you are using SSH configured to redirect traffic from the local port 4444 to port 80 of the example.com machine you are auditing, you can use the following command to make VHostScan connects through your localhost:4444 SSH tunnel, but make the request headers suitable for a direct connection to port 80:

|  |  |
| --- | --- |
| 1 | VHostScan -t localhost -b example.com -p 4444 -r 80 |

## Conclusion

### [VHostScan](https://github.com/codingo/VHostScan) help

Usage:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | VHostScan [-h] -t TARGET\_HOSTS [-w WORDLISTS] [-b BASE\_HOST] [-p PORT]                   [--prefix PREFIX] [--suffix SUFFIX] [-r REAL\_PORT]                   [--ignore-http-codes IGNORE\_HTTP\_CODES]                   [--ignore-content-length IGNORE\_CONTENT\_LENGTH] [--first-hit]                   [--unique-depth UNIQUE\_DEPTH] [--ssl] [--fuzzy-logic]                   [--no-lookups] [--rate-limit RATE\_LIMIT] [--waf] [-v]                   [-oN OUTPUT\_NORMAL | -oJ OUTPUT\_JSON | -oG OUTPUT\_GREPABLE]                   [--random-agent | --user-agent USER\_AGENT] |

Optional arguments:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47 | -h, --help            show this help message and exit  -t TARGET\_HOSTS       Set a target range of addresses to target. Ex                        10.11.1.1-255  -w WORDLISTS          Set the wordlists to use (default ./wordlists/virtual-                        host-scanning.txt)  -b BASE\_HOST          Set host to be used during substitution in wordlist                        (default to TARGET).  -p PORT               Set the port to use (default 80).  --prefix PREFIX       Add a prefix to each item in the word list (dev, test                        etc)  --suffix SUFFIX       Add a suffix to each item in the word list  -r REAL\_PORT          The real port of the webserver to use in headers when                        not 80 (see RFC2616 14.23), useful when pivoting                        through ssh/nc etc (default to PORT).  --ignore-http-codes IGNORE\_HTTP\_CODES                        Comma separated list of http codes to ignore with                        virtual host scans (default 404).  --ignore-content-length IGNORE\_CONTENT\_LENGTH                        Ignore content lengths of specificed amount (default                        0).  --first-hit           Return first successful result. Only use in scenarios                        where you are sure no catch-all is configured (such as                        a CTF).  --unique-depth UNIQUE\_DEPTH                        Show likely matches of page content that is found x                        times (default 1).  --ssl                 If set then connections will be made over HTTPS                        instead of HTTP (default http).  --fuzzy-logic         If set then fuzzy match will be performed against                        unique hosts (default off).  --no-lookups          Disable reverse lookups (identifies new targets and                        appends to wordlist, on by default).  --rate-limit RATE\_LIMIT                        Amount of time in seconds to delay between each scan                        (default 0).  --waf                 If set then simple WAF bypass headers will be sent.  -v                    Print verbose output  -oN OUTPUT\_NORMAL     Normal output printed to a file when the -oN option is                        specified with a filename argument.  -oJ OUTPUT\_JSON       JSON output printed to a file when the -oJ option is                        specified with a filename argument.  -oG OUTPUT\_GREPABLE   Grepable output printed to a file when the -oG option                        is specified with a filename argument.  --random-agent        If set, then each scan will use random user-agent from                        predefined list.  --user-agent USER\_AGENT                        Specify a user-agent to use for scans |