

---

# An introduction to information security using DNS vulnerabilities

**Robin M Snyder**  
**robin@robinsnyder.com**  
**http://www.robinsnyder.com**

## Abstract

Networking is for sharing resources. Security is for insuring that resources are not shared too much. The DNS, Domain Name System, converts names into IP addresses and, without it, the Internet would not work the way it does. Recent DNS security vulnerabilities have caused concern in the Internet community. This paper/session will introduce some fundamental principles of security, ways to keep up-to-date on security, and some current and common security issues and solutions, all done in terms of how DNS works, sometimes does not work, and DNS security vulnerabilities and how they impact the end user.

## Introduction

This paper/session introduces some fundamental principles of security, ways to keep up-to-date on security, and some current and common security issues and solutions, all done in terms of how DNS works, sometimes does not work, and DNS security vulnerabilities and how they impact the end user.

## Background

The Internet works on IP (Internet Protocol) addresses but people work better with domain names such as **amazon.com**, **audible.com**, etc. A DNS (Domain Name System) server provides a way to convert domain names into the associated IP (Internet Protocol) address. The DNS is a distributed hierarchical (tree-like) naming system database that removes the requirement to create and maintain a centralized database of names and IP addresses. The following are some top-level domain extensions.

- .com
- .edu
- .gov
- .mil

A reverse lookup can also be done. That is, find one or more domain names associated with an IP address.

In Windows, the **nslookup.exe** command provides a simple and quick command-line approach to doing DNS and reverse-DNS queries.

Command:

```
nslookup robinsnyder.com
```

Output:

```
Server: google-public-dns-a.google.com
Address: 8.8.8.8

Non-authoritative answer:
Name:   robinsnyder.com
Address: 69.51.22.77
```

A domain name may have more than one associated IP address. Large domains such as **microsoft.com** need more than one IP address for a domain name to avoid bandwidth congestion through a single IP and to redirect traffic when subject to DOS (Denial of Service) attacks.

Here is a command for a reverse-DNS lookup

```
nslookup 69.51.22.77
```

Output:

```
Server: google-public-dns-a.google.com
Address: 8.8.8.8

Name:   bellsdriving.com
Address: 69.51.22.77
```

There may be more than one domain name assigned to an IP address. An ISP (Internet Service Provider) may use one IP address to handle many domains. The domain requested is part of any request to that IP address so that the server at that address can route the request to the appropriate server. Note that a server can be a piece of hardware running server software or one of many processes running server software on the same piece of hardware.

In Windows, each computer connected to the Internet has two DNS servers specified by IP. Question: Why will a domain name not work? Answer: One should not assume a solution to the problem being solved. A domain name would need to be looked up, but that is what the DNS server is for, so an IP is needed.

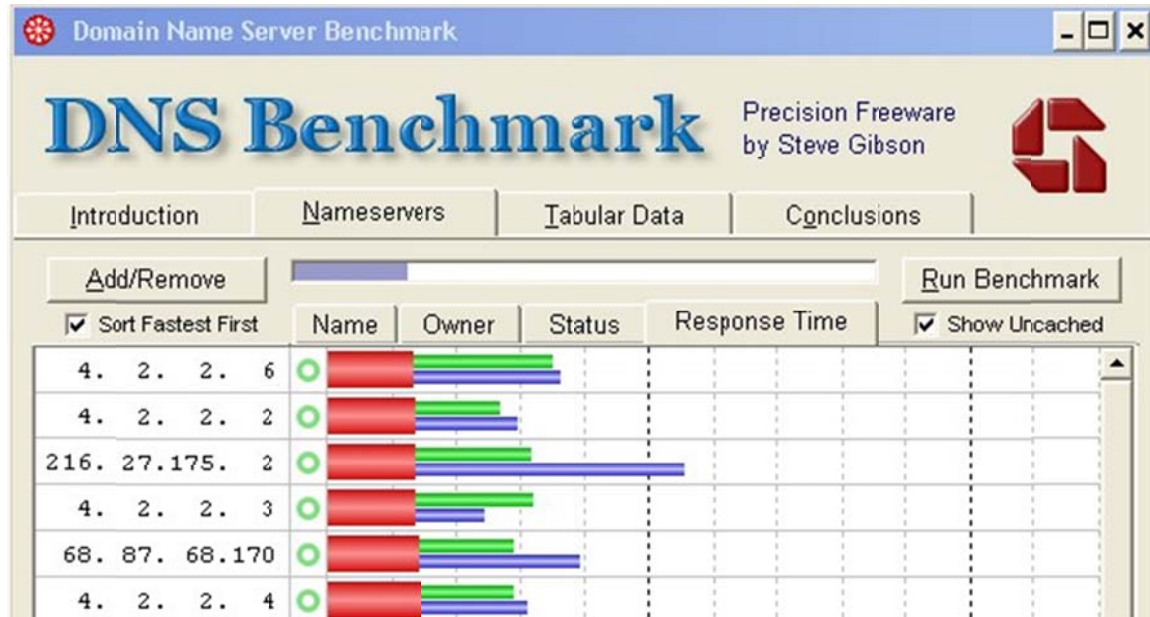
In Windows, DNS servers for a resource can be automatically obtained from a local resource using, for example, DHCP (Domain Host Configuration Protocol) or the DNS servers can be explicitly specified by the administrator of the resource.

Some popular DNS servers are the following.

- Google at **8.8.8.8** and **8.8.8.4**
- OpenDNS at **208.67.222.222** and **208.67.220.220**

An internet search for "**public DNS servers**" will return a list of many public DNS servers.

An ISP (Internet Service Provider) will often run their own DNS server. This may or may not be the best solution for a given user. A good well-known and free benchmark program is the "**DNSBench**" program from Gibson Research Corporation. Written by security expert Steve Gibson, author of the disk recovery tool Spin-Rite, DNSBench can be downloaded from <http://www.grc.com/dns/benchmark.htm>. Here is how a typical result might appear.



Some DNS servers will display advertising when a domain name cannot be located. Some DNS servers, such as OpenDNS, may convert simple misspellings into the possible intended domain name, such as converting **amazo.com** to **amazon.com**. Some offer choices when a domain name is not found.

The more popular and used a DNS server is, the more likely it is to be both fast (from caching, etc.) and secure (from having the latest patches).

OpenDNS provides accounts whereby one can restrict and control access to domains. This can be useful for, say, parent administrators who wish to restrict Internet access of their children on family computers.

If the DNS server is not working, the user can always use the IP address directly instead of the name. To do this, type the IP address instead of the domain name.

Location:

Knowing the IP of a domain name and entering it into a web browser may be of limited use in circumventing such domain name blocking unless the entire web site uses relative references and works as desired with just an IP address.

Another way to control access of domain-to-IP lookup on a Windows computer is to use the **hosts** file.

### Hosts file

In Windows, the hosts file is used to map host names to IP (Internet Protocol) addresses at the local machine level. It is a text file that provides a way convert domain names to IP addresses for

the machine on which the hosts file is located. The **hosts** file in Windows provides some flexibility and some security problems. The **hosts** file location varies by Windows operating system, such as the following Windows XP default location.

```
C:\WINDOWS\SYSTEM32\DRIVERS\ETC\
```

A sample **hosts** file is called **hosts.sam** (for sample) and is located in the same directory as the **hosts** file.

Here is an example hosts file.

```
# my hosts file
127.0.0.1 localhost
168.20.203.67 www.mybank.com
127.0.0.1 ad.doubleclick.net
# ... all other mapped domain names ...
```

Note that the character "#" at the start of a line causes that line to be ignored. For example, the domain name **ad.doubleclick.net** has been mapped to the **localhost** IP address of **127.0.0.1**.

There are Internet sites that will provide lists of thousands of web sites that you probably do not want to visit nor do you want such hacking or marketing sites collecting personal information about you or your computer. You can download and install a text file from one of those sites to restrict access of your computer to marketing sites, etc.

The hosts file can be a major security problem as any program that can change the hosts file can redirect you to their site. For example, the above host file causes any request for **www.mybank.com** to go to my web site.

### Security

How does the user know that the server is who the server claims to be? For example, when the URL **http://www.company.com** is typed and **Enter** is pressed, how do you know that you have reached the company who owns that URL. The DNS (Domain Name System) is a distributed database system that is responsible for converting domain names, as used in web URL's, into IP addresses. It is possible, and it has happened, that a hacker could do the following.

- Copy a companies web site from the Internet and modify it on their own site as a Trojan horse web site.
- Break into one of the primary DNS computers and have the domain name point to their Trojan horse web site. Another, perhaps easier, way is to call the right office and use what is called social engineering to talk someone into making the switch.
- Now, the Trojan horse web site can pose as the real web site, at least until the company whose web site has been hijacked finds out and takes some form of effective action.

### DNS interrogation

DNS interrogation can be used

- to determine the IP address for a given domain name, or
- to determine one or more domain names for a given IP address.

Linux/Unix systems:

```
host 2.0.0.127.b.barracudacentral.org
2.0.0.127.b.barracudacentral.org has address 127.0.0.2
```

Windows systems:

```
nslookup 2.0.0.127.b.barracudacentral.org
Server: {Your DNS server hostname}
Address: {Your DNS server IP addresses}

Non-authoritative answer:
Name: 2.0.0.127.b.barracudacentral.org
Address: 127.0.0.2
```

## Blacklisting

About 90% of Internet email traffic is SPAM. A white-list is a sender IP address that is considered not to be SPAM. A black-list is a sender address that is considered to be SPAM. A quick blacklist test can be obtained from sites such as Spamhaus.

Sites such as Spamhaus collect anti-spam complaints and/or then make spammer IP addresses or domain names available. Email servers and/or clients that use these sites may then block email from those sites. Two common identifying attributes provided by anti-spam sites are domain names and IP (Internet Protocol) addresses.

A DNSBL (DNS-based Blackhole List), or block list, or blacklist, is a list of IP addresses published via the Internet DNS. A black list is a list of email domains or addresses that are to be considered spam. Many black lists available on the Internet are called RBL (Realtime Black List) which allows the list to be dynamically modified. Clients using the service will contact the service periodically to update their black lists - much in the same way as anti-virus software is updated periodically for new virus definitions.

These sites do not block email directly. Rather, email servers that use these services may decide to block email because of the presence of a domain or IP address on a list.

There are sites that will check many different RPL services and provide information as to which services have a IP address black-listed.

Here is an example of a link in an email message purporting to be "**Amazon.com Deal of the Day**".

```
http://84489.standbegan.ru/?47846
```

The first step is to find the IP for the domain name of **standbegan.ru**.

```
nslookup standbegan.ru
Server: google-public-dns-a.google.com
Address: 8.8.8.8

Non-authoritative answer:
Name: standbegan.ru
Address: 61.136.59.69
```

The next step is to query a RBL using DNS and interpret the results.

There are many web sites that allow such checking, such as the one at <http://www.spameatingmonkey.com>.

**Spam Eating Monkey**

Home Lists Usage Lookup Remove Contact FAQs Service

**Lookup a domain or IP**

What domain or IP would you like to lookup?

**Results**

<a href="#">SEM-FRESH</a>	Not listed
<a href="#">SEM-FRESH10</a>	Not listed
<a href="#">SEM-FRESH15</a>	Not listed
<a href="#">SEM-URI</a>	Black listed <a href="#">Request removal</a>
<a href="#">SEM-URIRED</a>	Red listed Black listed <a href="#">Request removal</a>

In this case, the IP was not blacklisted but the domain name was blacklisted. This might mean that the same domain name is being used but moved from IP to IP.

The web site text for SEM-URI is "# URIs found a requisite number of times in unwanted messages. Domains automatically expire after 15 days of inactivity."

The web site text for SEM\_URIRED is "URI early and preemptive detection list that lists URIs before they appear in the flow of unwanted mail. Domains automatically expire after 30 days of inactivity. This zone also includes all of SEM-URI."

Sites such as **spameatingmonkey.com** provide documentation on how to incorporate their site into SPAM tagging systems such as SpamAssassin.

The following is how to obtain the above results using `nslookup.exe`.

```
nslookup standbegan.ru.uribl.spameatingmonkey.net
Server: google-public-dns-a.google.com
Address: 8.8.8.8

Non-authoritative answer:
Name: standbegan.ru.uribl.spameatingmonkey.net
Address: 127.0.0.2
```

Note how the IP or domain name is prefixed to the DNS server doing the lookups. In this case, the DNS server to be queried (though the regular DNS server) is `uribl.spameatingmonkey.net` while the domain name to be checked is `standbegan.ru`.

There is no standard for how to return the results so the documentation for each web site returning blacklisting information must be checked. Many such sites return the local host address `127.0.0.1` where the last number is modified to be the result code. So, for the above results, `127.0.0.2`, the returned result is `2` which means that that domain or IP is blacklisted.

## DNS lookup

Obviously these methods become much more useful when the process is automated. If a large number of queries are to be done, thread programming becomes more and more important in obtaining faster overall results. See, for example, (snyder, 19154).

One way to do a DNS lookup and a reverse DNS lookup is demonstrated by the following Python program.

```
# Purpose: Demonstrate DNS lookup

import socket

print "Host to IP to IP to host:"
for host1 in ["www.amazon.com", "www.acm.org"]:
    ip1 = socket.gethostbyname(host1)
    try:
        host2, alias2, ip2 = socket.gethostbyaddr(ip1)
    except:
        host2 = "not found"
        ip2 = [ip1]
    print "host=" + host1
    print " -> ip=" + ip1
    print " -> ip=" + str(ip2)
    print " -> host=" + host2
```

Here is the output.

```
Host to IP to IP to host:
host=www.amazon.com
-> ip=72.21.206.5
-> ip=['72.21.206.5']
-> host=206-5.amazon.com
host=www.acm.org
-> ip=63.118.7.16
-> ip=['63.118.7.16']
-> host=acm25-7.acm.org
```

Note how the assignment statement supports extracting the results of a function that can return more than one result (e.g., as a list). Note that there may be more than one host name that maps to the same IP address. In some cases, a reverse lookup of IP to host may not work. The `try/except` construct can be used to handle such exceptions.

## IP to country

An interesting use of DNS is to access a DNS server that provides an IP to country lookup. Such a DNS server works in a manner similar to the DNS servers providing services for blacklist, whitelist, etc. Here is how **spameatingmonkey.net** does it to determine if an IP is outside the country.

```
nslookup 61.136.59.69.us.country.spameatingmonkey.net
Server: google-public-dns-a.google.com
Address: 8.8.8.8

Non-authoritative answer:
Name: 61.136.59.69.us.country.spameatingmonkey.net
Address: 127.0.0.2
```

Keep in mind that anyone can use an anonymizing service such as Tor (or other proxy server) so that the IP one sees may or may not be where the originating IP is located.

## Security issues

Technically, the DNS protocol uses UDP (User Datagram Protocol) rather than TCP (Transmission Control Protocol). TCP is reliable, cannot be spoofed, insures delivery, but has more overhead - and not used for that reason. UDP has less overhead and is a try and see approach to data communication. The basic issue arises in that UDP can be spoofed. That is, a DNS server has no way of knowing, for sure, from where the packet originated.

## BIND

DNS security is related to the DNS server software being used. BIND is a popular open source DNS protocol system maintained by the ISC (Internet Systems Consortium).

*BIND is open-source software that implements the Domain Name System (DNS) protocols for the Internet. It is a reference implementation of those protocols, but it is also production-grade software, suitable for use in high-volume and high-reliability applications.*  
<https://www.isc.org/software/bind> [as of Wed, Aug 26, 2009]

## Summary

This paper/session has presented some fundamental principles of security, ways to keep up-to-date on security, and some current and common security issues and solutions, all done in terms of how DNS works, sometimes does not work, and DNS security vulnerabilities and how they impact the end user.

## References

- [160] Snyder, R. (2007). Security programming using Python: Man-in-the-middle attacks 2007 Information Security Curriculum Development conference (September 28-29, 2007), Kennesaw, GA. CD.
- [159] Snyder, R. (2007). Introducing Python programming into the data communications and security courses 2007 Information Security Curriculum Development conference (September 28-29, 2007), Kennesaw, GA. CD.
- [154] Snyder, R. (2007). Simple security programming for students using Portable Python 1st Computer Security Conference (April 12-13, 2007), Myrtle Beach, SC. CD.
- [99] Snyder, R. (2002). The fundamentals and practical use of certificate-based security in secure web-based systems 35th Annual Conference of the Association of Small Computer Users in Education (June 9-13, 2002), Myrtle Beach, SC. pages 223-234.
- [15] Snyder, R. (1994). Proactive approaches to information systems and computer security 27th Annual Conference of the Association of Small Computer Users in Education (June 12-16, 1994), Myrtle Beach, SC. pages 236-242.