

An improved protocol for the examination of rogue WWW sites

Angus M. Marshall BSc CEng MBCS FRSA
Centre for Internet Computing
University of Hull
Scarborough Campus
Filey Road
Scarborough YO11 3AZ
a.marshall@hull.ac.uk

July 9, 2003

Abstract

Internet server managers have a range of techniques available to help them improve service performance and security. These techniques can become barriers to the investigation of illicit or illegal activity.

This paper describes some of the legitimate techniques which can be used to improve server performance or security, and which present challenges for the investigator. Furthermore, it proposes a rigorous procedure which should be followed to ensure that any investigation of a web site or server has been complete and accurate, and that all possible useful information has been extracted and examined.

Acknowledgements

The author is indebted to the team at the Digital Evidence Recovery and Internet Crime (DERIC) unit of North Yorkshire County Council Trading Standards Dept. for their assistance with the detail of current investigatory techniques and also for providing a good example of one of the masking techniques discussed in this paper.

He also wishes to thank Brian Tompsett for his advice & support and the reviewers who commented on this paper for their helpful comments.

Disclaimer

Throughout this paper, examples are given of techniques which can be abused. The majority of examples (with the exception of www.deric.org.uk) are taken from test environments configured, by the author, on standard software. All examples used are fictitious but based on real configurations.

1 Introduction

Many criminals are turning to the internet as a means of protecting their identity while improving their access to victims and materials. The majority of these crimes fall into the “Internet-assisted” and “Internet-enabled” categories within Marshall & Tompsett’s taxonomy [1].

The most common use of the internet is as a communications medium to contact associates or victims and, as a result, the most fruitful source of evidence still lies in the realm of the “static” data held on the computer used to originate or receive messages and/or data involved in the criminal activity.

A small number of criminals, however, are turning to technologies such as the World-Wide-Web [2] to assist with, or as the sole means of, commissioning criminal activities.

This paper concentrates on the investigation of WWW sites. but it is believed that the techniques described herein may be adapted to the investigation of other internet services.

2 Current Procedures for Investigation of rogue WWW sites

2.1 Site inspection

Traditionally, when a rogue WWW site is suspected, the investigator’s first instinct is to launch a web browser and gain access to the site themselves to check the content of it. Once they are satisfied that the content is of such a nature that action should be taken, they perform some standard checks to locate the server hosting the site and the persons responsible for maintenance of the domain.

As we shall see later this approach is not sufficient as the investigator may, in fact, be examining the wrong WWW site altogether, despite it having the same fully qualified domain name (FQDN) as the suspect site.

2.2 Traceroute

Using the traceroute [3] command (tracert in Microsoft operating systems) the investigator can perform a complete network trace (Fig. 1) from the location of the tracing machine to the server apparently hosting the suspect domain. More recently, several graphical tools have come into common usage. These attempt to plot the geographical location of each node onto a map.

In most cases, the route tracing program relies on UDP and ICMP to interrogate intermediate network nodes between the tracing machine and the target. This can be done by repeatedly sending UDP datagrams from source machine to target machine with successively incremented TTL¹ fields.

Each node on the route decrements the TTL field as the packet passes through it. If the TTL field reaches 0, the decrementing node will report an ICMP “TIME_EXCEEDED” (more correctly, maximum hops exceeded) error

¹TTL in traceroute refers to the “hop-count” or number of intermediate nodes through which the packet moves from source to destination, not the time taken.

```

traceroute to www.deric.org.uk (209.207.188.198), 30 hops max, 38 byte packets
 1 biscotti.cicnet (172.16.8.3) 35.206 ms 0.676 ms 0.670 ms
 2 150.237.7.1 (150.237.7.1) 2.473 ms 2.263 ms 2.286 ms
 3 150.237.254.253 (150.237.254.253) 10.915 ms 15.837 ms 14.365 ms
 4 hwall1-gw.ucc.hull.ac.uk (150.237.254.254) 9.856 ms 6.343 ms 5.562 ms
 5 * 150.237.35.1 (150.237.35.1) 5.934 ms 5.848 ms
 6 * a0-5.leeds.yhman.net.uk (195.195.130.141) 7.756 ms 8.394 ms
 7 leeds-bar.ja.net (146.97.40.13) 7.458 ms 10.874 ms 7.499 ms
 8 po9-0.leed-scr.ja.net (146.97.35.69) 7.684 ms 8.013 ms 7.584 ms
 9 po3-0.lond-scr.ja.net (146.97.33.70) 12.341 ms 15.255 ms 11.925 ms
10 po2-0.london-bar5.ja.net (146.97.35.138) 11.607 ms 12.506 ms 12.420 ms
11 pos3-0.gw1.lnd9.alter.net (158.43.37.201) 11.710 ms 12.359 ms 11.698 ms
12 so-3-0-0.xr2.lnd9.alter.net (158.43.150.145) 11.892 ms 16.584 ms 16.118 ms
13 so-1-1-0.TR1.LND9.Alter.Net (146.188.15.37) 11.945 ms 12.756 ms 11.924 ms
14 so-2-0-0.IR2.NYC12.Alter.Net (146.188.8.178) 83.909 ms 83.498 ms 83.368 ms
15 0.so-0-0-0.IL2.NYC9.ALTER.NET (152.63.23.65) 83.696 ms 83.627 ms 83.623 ms
16 0.so-3-0-0.TL2.NYC9.ALTER.NET (152.63.9.186) 84.938 ms 83.594 ms 83.539 ms
17 0.so-1-0-0.XL2.NYC9.ALTER.NET (152.63.23.130) 83.636 ms 83.700 ms 83.595 ms
18 POS7-0.BR2.NYC9.ALTER.NET (152.63.22.229) 83.486 ms 83.436 ms 83.527 ms
19 a3-0-2-0.r01.nwrknj01.us.bb.verio.net (129.250.9.165) 84.661 ms 84.521 ms 84.260 ms
20 p16-1-1-1.r21.nycmy01.us.bb.verio.net (129.250.5.13) 108.884 ms 85.800 ms 85.721 ms
21 p16-5-0-0.r02.asbnva01.us.bb.verio.net (129.250.5.99) 92.661 ms 93.103 ms 92.667 ms
22 p16-1-0-0.r02.mclnva02.us.bb.verio.net (129.250.2.180) 92.636 ms 92.671 ms 93.106 ms
23 p16-0-0-0.r01.mclnva02.us.bb.verio.net (129.250.5.253) 91.604 ms 94.166 ms 91.469 ms
24 p4-9-0.a00.alxnva02.us.da.verio.net (129.250.17.58) 91.562 ms 92.061 ms 91.967 ms
25 ge0031.ed1.wdc.dn.net (216.167.88.116) 92.227 ms 92.375 ms 92.171 ms
26 209.207.188.198 (209.207.188.198) 92.069 ms 92.973 ms 97.388 ms

```

Figure 1: Text-based traceroute showing intermediate nodes and timing information

to the source. In this way the source finds the first, second and subsequent nodes on the route through repeated probing.

At the target machine the recipient identifies the “illegal” port used in the UDP datagram and reports an ICMP “Unreachable port” message.

This process itself may be sufficient to alert the criminal that an investigation is under way, although given the volumes of traffic present on the internet this is unlikely. Generally speaking though, it is rare for a traceroute to be used, except when there is a suspicion of network congestion or failure, and even in these cases, network administrators, tend only to interrogate machines that they know should be interacting directly with their network.

The likelihood of success of a traceroute is highly variable, in any case, as many network providers now block or redirect (again using ICMP) UDP traffic as a matter of course as it can be used to mount Denial of Service attacks on their servers. For this reason, care should be taken to ensure that a single traceroute is not relied upon. Alternative traceroutes using TCP (connection-oriented) packets and specified ports (e.g. port 80 for WWW server tracing) are available.

For the sake of completeness, a traceroute or similar should certainly be attempted by the investigator, and repeat probing at intervals may be appropriate to determine if the site under investigation is “moving” (i.e. changing it’s IP address at intervals) in an attempt to render examination more difficult. Care should be taken however, to ensure that the suspect is not alerted to the investigative process. The most likely cause of suspicion on the part of the alleged miscreant will be the appearance of repeated probes in their log files, all from the same IP address or network. Therefore, if traceroute probing is to be used, the investigator should use a range of IP addresses, preferably on different networks. These can easily be obtained by setting up a number of accounts with different ISPs. In all cases, the use of IP addresses associated with regulatory networks such as “police.uk” and “gov.uk” is strongly discouraged as the appearance of these names in a log file may immediately suggest not only that an investigation is under way, but also alert the suspect to the exact source of the investigation.

If the traceroute is successful, the investigator can often obtain the apparent geographical location of the target machine by obtaining geographical data about all nodes on the route. This information, however, should not be considered definitive. Any geographical information provided about an IP address is dependent upon it being volunteered by the “owner” of the address either through RFC1876 [4] compliant “LOC” records in DNS or via registration with a location database associated with a particular tool (e.g. McAfee Visual Trace (formerly Neotrace) [5]).

However, some information about IP addresses and FQDNs may be considered trustworthy because

1. historically, IP address blocks were assigned geographically to providers
2. IP address block allocations are registered in publicly available databases
3. ISPs tend to name their equipment according to a scheme in which the location of the equipment forms part of the name (e.g. SPRINT-NY-POP-3 might be Sprint’s 3rd Point of Presence in New York) or is easily traceable by some other means.

It is important to remember, that the top-level domain suffix of a site (e.g. .uk, .tv, .ru) is not a valid indicator of the location of the server. Anyone, anywhere in the world, can currently register a domain within any of the TLDs, providing they are willing to pay the appropriate fee. Additionally, new TLDs being issued and proposed are non-geographic in nature.

2.3 WHOIS databases

In addition to attempting to determine the approximate geographical location of the suspect server, the investigator can attempt to determine the owner of the domain name responsible for the suspect activity. This may be performed independently of the traceroute. The public WHOIS databases maintained by the internet registrars contain standard information for every domain registered on the internet. Figure 2 shows a sample record. Again, the information contained in the WHOIS record may not be definitive. It is common practice for some WWW hosting providers and domain registration services to register domains on behalf of clients without disclosing the clients' details to the registrar. This allows them to retain control of the domain and prevents the client transferring to another hosting provider easily.

Furthermore, local legislation (e.g. the UK Data Protection Act(1998) [6]) may prevent full disclosure of all ownership details unless appropriate legal authority has been obtained, particularly where the domain has been registered by a private individual rather than a company or other trading entity.

3 Pitfalls

The techniques outlined above, while obviously subject to some severe limitations, can return a considerable amount of useful information, but they fail to take account of a number of approaches that webmasters use to improve server performance, improve response time, and enhance security. Amongst these techniques are :

1. DNS clustering
2. Mirroring
3. Load Balancing
4. Domain Forwarding, Redirection and Masking
5. Intermediate Proxies

Each of these presents a challenge for the investigator as it effectively increases the quantity of investigative work required by duplicating and relocating the content of a WWW site in many different servers, possibly located in different geographical zones.

To appreciate the additional investigative work required, it is useful to summarise each of these approaches and its legitimate use along with a note of how it may be (ab)used.

[whois.nic.uk]

Domain Name:
deric.org.uk

Registrant:
North Yorkshire County Council Trading Standards
Trading As: North Yorkshire County Council Trading Standards

Registrant's Address:
Standard House
48 High Street
Northallerton
North Yorks
DL78EQ

Registrant's Agent:
Fibranet Services Ltd [Tag = FIBRANET]

Relevant Dates:
Registered on: 18-Feb-2002
Last updated: 02-Oct-2002

Name servers listed in order:
ns3.ukdnsservers.co.uk 161.58.88.37
ns4.ukdnsservers.co.uk 216.167.70.1

WHOIS database last updated at 15:00:02 21-Jan-2003

--

(c) Nominet UK

For further information and terms of use please see <http://www.nic.uk/whois>
Nominet reserves the right to withhold access to this service at any time.

Figure 2: Sample WHOIS record for deric.org.uk retrieved from the whois.nic.uk server

3.1 DNS clustering / round-robin DNS

Within the domain name system (DNS) [7] a single FQDN may be mapped to multiple Internet Protocol (IP) addresses through the use of address (A) or alias (CNAME) records (See figure 3). When DNS servers are queried to provide the mapping from human-readable FQDN to machine-usable IP, they typically provide responses in round-robin fashion. i.e. the first enquirer receives the first address entry, the second enquirer receives the second address entry and so on until the DNS server's list is exhausted and it restarts at the top of the list (See figure 4).

```
@ IN SOA gallifrey.cicnet. root.gallifrey.cicnet. (
    2002080801 ;
    21600 ;
    1800 ;
    604800 ;
    86400 ) ;

    IN NS  gallifrey.cicnet.

;Primary Servers
localhost IN A 127.0.0.1
gallifrey IN A 172.16.8.4
ns1 IN CNAME gallifrey.cicnet.
; Round-robin DNS entries for a simple load-balancer
real IN A 172.16.8.2
      IN A 172.16.8.254
```

Figure 3: Sample DNS record for a small round-robin based cluster. Note that “real.cicnet” has been associated with two IP addresses

```
Server: 172.16.8.4
Address:      172.16.8.4#53

Name: real.cicnet
Address: 172.16.8.254
Name: real.cicnet
Address: 172.16.8.2
```

Figure 4: Retrieval of complete DNS entries for a FQDN. Note that two IP addresses are returned for the FQDN “real.cicnet”

Thus, where round-robin based DNS clustering is in use, the investigator is unlikely to be examining exactly the same server as the person reporting suspect activity. Furthermore, since most end-user client software caches DNS information locally for the duration of a session simply repeatedly accessing the same site without closing the browser is insufficient to ensure that all instances of the FQDN have been examined.

3.2 Mirroring

For DNS clustering to be effective, the webmaster must implement a replication [8] or mirroring mechanism to keep all instances of the “server” in step with a master copy. Because these mirror sites are independent entities containing copies of the master site, an opportunity exists for one or more instances to be “out of step” with the master copy, or for one or more to contain information which is not shared with the other instances for some reason.

Hence, when investigating the content of a site it is important that all copies are thoroughly examined to locate either old, undeleted, evidence or unique content which has been hidden amongst legitimate content on only a few of the servers.

An example of such a situation, where replication and deliberate non-replication are mixed, would be the case of a multi-national organisation which maintains a “corporate” WWW presence mirrored to servers at the head office for each country. Each head office could also maintain a separate, local, WWW area to provide country-specific information directly from the appropriate server.

It should also be noted, at this point, that using “webget” type software, which attempts to retrieve all pages on a site, is not sufficient for a complete investigative retrieval. This software relies on following links from the start page to find all linked pages. Thus it will miss any “hidden” pages which are not directly linked to others in the site. Hence, it cannot be guaranteed that any copies of the site thus produced are forensically sound. Equally, where site content is generated dynamically, such software may not be able to produce the full range of interactions required to fully explore the site (some sites now regularly deploy such anti-robot measures as requiring a user to type in a randomly generated string, displayed as an image [9], to confirm that the user is a human being instead of a software agent).

Instead, attempts must be made to determine the full contents of the site by manual exploration and by probing for possible hidden files and directories.

The first source of information about the structure of a site is the HTML used to make up the pages themselves. Inside this HTML there will be a range of tags containing pointers and links to other files used in the page and the site. These will contain either absolute references (full URLs ²) or relative references (partial URLs ³).

Full URLs, particularly in “IMG” and related tags such as those relating to included files (e.g. CSS and Javascript) indicate that the site may have been spread across multiple servers for some reason. In this case, the investigation must expand to encompass the additional “sites” referenced in these tags.

Partial URLs, on the other hand, disclose considerable information about the logical structure of the site (i.e. the view that is presented to the end-user). This logical structure is often an accurate representation of the physical structure used to hold the site on the physical server, although this is not guaranteed for reasons which will be explained later.

The “robots.txt” file used to control search-engine spidering activity [10] may be beneficial here, if it is present. Illicit content in a hidden file or directory may be marked as non-indexable in the “robots.txt” file to prevent it becoming

²conforming to the standard “protocol://user:password@server:port/resource” format (e.g. <http://thissite.com/nextpage.html>)

³usually containing just a directory path and filename or similar (e.g. [images/picture1.gif](#))

accidentally visible through search engines. This file can also prevent some “webget” software (e.g. WebReaper) from collecting a full image of the site.

Thus a valid first step would be to attempt to retrieve a “robots.txt” file from all directories discovered on the site. The “meta” tags [11] in all pages should also be checked to see if they contain “robot” constraints of a similar type. It should be noted, of course, that the “robots.txt” file, and its equivalent “META” tag, are not a requirement on any WWW site, nor is their presence an absolute requirement to prevent spidering of the files and directories on a site. Spiders generally only follow the links found on a site, so if the illicit content is not the target of a link in a page, it should not be indexed anyway.

It is also possible that, due to poor HTML coding or configuration of the server, the “entry” page to the WWW site has a filename which is not the same as the default page to be presented by the server. In this case, if the default page is not present, many servers will offer a listing of directories at the root level for the site being viewed. A URL under investigation, may end in either a “/” (e.g. <http://www.cic.hull.ac.uk/>), in which case the server is expected to deliver a default page, or a resource name (e.g. <http://www.cic.hull.ac.uk/index.html>) where the user is required to specify the exact resource they require. In the former case, the default page is likely to be present (although this may be a server-generated listing by design) whereas in the latter the removal of the resource name may allow the server to disclose additional information about the site structure.

If all else fails, a brute-force retrieval attempt using a filename generator may be attempted, but this will take a considerable amount of time (determined by the length of the longest filename supported by the server) and may, indeed, cause the server being probed to crash or exceed its allocated bandwidth limits, causing it to disappear from the internet for an indeterminate time.

In this case, it is highly likely that the site owner will discover that an attempt has been made to retrieve the hidden information (not least because of the sudden increase in the number of 404 errors [12] they are recording in their server logs), and they may take measures to remove the illicit material before any real evidence of its existence on the server has been recovered.

Any files that are recovered should be cross-referenced to ensure that they are, indeed, being used in the pages which are under investigation. Many sites accumulate “entropic clutter” as they evolve and old images, HTML etc. can often be left for several years without being used actively in any real parts of a site. Unused files should not be completely discounted though, as they may be located on the server for easy download but not linked to any pages, creating a “blind site” which is notionally only accessible by knowledgeable users.

Techniques involving other protocols (e.g. using ftp to attempt to gain access to the server filesystem directly) have been suggested, the legality and advisability of such techniques is questionable.

The use of standard HTTP to recover file system information is relatively forensically sound as it is very easy to copy data from the server but difficult to transfer data to the server without the cooperation of additional programs on the server. Where other protocols, particularly ftp, are used, there is a danger that the investigator can lay themselves open to the challenge that they have introduced new material to the server, and thus compromised the investigation.

Furthermore, if a site is expected to be normally accessed using only HTTP, the appearance of non-HTTP session in the server log files may act as a warning

to the suspect again.

3.3 Dynamic content

The discussion above tends to assume that the content of a site is largely composed of static HTML pages, however modern techniques are moving away from this and placing more emphasis on dynamically generated pages, where programs on the server write web pages on demand, often integrating information from a variety of sources. Where this is the case, the investigator is unlikely to be able to retrieve an accurate copy of the “raw” site, but should be able to gather an accurate representation of the “logical” site as it is presented to the user. Warning signs that dynamic generation is in progress can usually be found in the URLs contained in the pages themselves, primarily in the endings of files names (e.g. .asp, .jsp, .php, .cgi, .pl, .dll etc.⁴). The presence of GET method data in URLs (normally trailing the filename as in “/fetch.cgi?page=&user=5”) is also indicative of server-side processing being used to control page creation.

The presence of server-side processing does not invalidate the investigative process, though, as the end users viewing the site in its “normal” mode of operation will also be using these server-side programs and hence will be presented with a similar logical view to that seen by the investigator.

3.4 Load balancing

Where a DNS-based cluster has been implemented, thorough examination is, comparatively, straightforward (see section 3.1) as all addresses are publicly available and the investigator can retrieve the sites using normal tools.

However, there are several products available in both hardware and software which provide a “load-balancing” mechanism associated with a single public IP address. In these cases, a load-balancer acts as an intermediary between a group of servers and the end-user. This load-balancer accepts all incoming requests and directs them, transparently, to an internal server for processing. The results from the internal server are then presented back to the load-balancer which forwards them to the end-user. Thus a large cluster of servers appears to be a single coherent entity.

Because the operation of the load-balancer is determined by the loads being placed on the servers it manages, and thus is dependent not only on the number of incoming connections, but also on the nature of each transaction taking place on the cluster, it is externally non-deterministic (i.e. it is impossible for an external observer (single user) to predict which server in the cluster their request will be directed to, as they have little or no knowledge of the activities of other sessions using the cluster concurrently). It should be noted that the DNS round-robin mechanism described above is completely deterministic as a knowledgeable user can elect to connect directly to a server by IP address rather than by FQDN, thus eliminating DNS from the process completely. This may require the use of HTTP/1.1 [12] to ensure that a correct virtual host is selected on the target server.

Opportunities for the hiding of information, by an external agent, on a single server in the cluster are minimised as it is difficult for an illicit end-user to

⁴These can all be redefined at will by the server manager, but it is rare for this to be done

specify which server within the cluster they wish to use. However, it is possible to compromise a single server in the cluster and then rely on the cluster's own replication or on viral propagation techniques to ensure that illicit content is distributed across the entire cluster.

If the person wishing to hide the information can operate "inside" the protected network located "behind" the load-balancer, they have a very good chance of being able to place unwanted information on a single server, but their incentive for doing so is minimised by the difficulties external users will face in trying to select the "infected" server noted above.

Clusters such as this still require mirroring, and opportunities exist for servers to become de-synchronised leading to a small possibility that old information can be retrieved. This requires more luck than skill as the selection of server by the load-balancer depends entirely on network and server load conditions at the time of access.

Generally speaking, however, if the content being sought or investigated is believed to originate from a load-balanced cluster, the investigator may treat the cluster as equivalent to a simple single server environment.

3.5 Domain forwarding

Many low cost domain name registration services offer, at the time of writing, a "domain forwarding" service which uses their DNS server and WWW server to allow users to connect to the subscribers web site via a URL which appears to be more "legitimate" than that provided by the site which is really hosting the WWW site. (e.g. <http://www.mysite.com/> has more internet "credibility" than <http://www.mysite.dodgyfreehosting.net/>).

The simplest method to implement this forwarding is to use the HTTP 30x series of responses to redirect the browser from the "legitimate" host to the real host. However, many subscribers perceive this as undesirable because the redirect often causes a warning to be issued to the browser, and in all cases will reveal the real host of the site in question.

For these reasons, amongst others, some WWW forwarding services offer a URL masking forwarding service which attempts to hide the real identity of the WWW site and thus enhance the image of "legitimacy" offered by the registered domain name.

3.6 Masking

URL masking can be performed in two ways, frame-based and reverse-proxy based. Of these two, the frame based solution seems to be the most common. As discussed below, this may not be the real case as reverse-proxy masking is extremely difficult to detect, whereas frame-based presents some very obvious clues for even the inexperienced investigator.

3.6.1 Frame-based URL masking

In frame-based URL masking, a HTML frameset is generated by the redirecting server. This frameset typically takes the form shown in Figure 5 This causes the browser to perform a two-stage process. Firstly, it retrieves the frameset code from the WWW sever reference by the FQDN. The frameset instructs it

to create a sub-window of the viewable window, in this case occupying 100% of the height of the viewable window, and to fill this sub-window with content retrieved from the URL referenced in `frame src` line. Thus, the user sees only the “legitimate” FQDN in the browser location box while the content is retrieved from a different location altogether (See Figure 6 for more detail).

```
<html>
  <head>
    <title>My dodgy WWW site.</title>
  </head>
  <frameset rows='100%'>
    <frame src='http://www.mysite.dodgyfreehosting.net/'>
  </frameset>
</html>
```

Figure 5: HTML frameset code and result

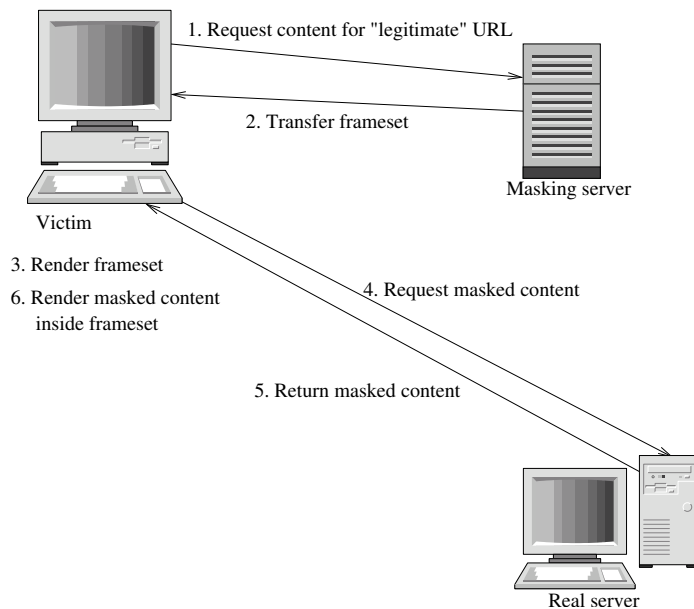


Figure 6: URL masking sequence of events

Detection of this form of masking is easy. The investigator, upon using the “view source” option in any common WWW browser will be presented with HTML similar to that in Figure 1 and can discover the real FQDN of the suspect site from the embedded URL.

3.6.2 Proxy based URL masking

Another approach to masking and load balancing involves the use of a reverse proxy to create an “invisible” conduit between the FQDN server and

the server(s) containing the real content.

Splitting the content of a site across multiple servers and using a reverse proxy provides some basic load balancing as the reverse proxy can amalgamate the different server identities together under a single FQDN.

3.6.3 Reverse proxies

Reverse proxying is a feature of the Apache WWW server [13] amongst others. In Apache, the facility is enabled using the `ProxyPass` and `ProxyPassReverse` features. In use, these instruct the Apache WWW server to operate in proxy rather than server mode and to translate HTTP header information as it passes through the proxy so that content may be retrieved from the proxied host and presented to the end-user as if it had come from the Apache server machine itself. This process (shown in Fig. 7) is completely transparent to the end-user and Apache adds no information to the HTTP headers to indicate that any translation or proxying has taken place. Hence it can be used to hide the real identity of a server. Figure 8 contain an example of the configuration, while Figures 9 and 10 show error headers from the real server and the reverse proxy respectively. The different version numbers of the two servers are clearly visible.

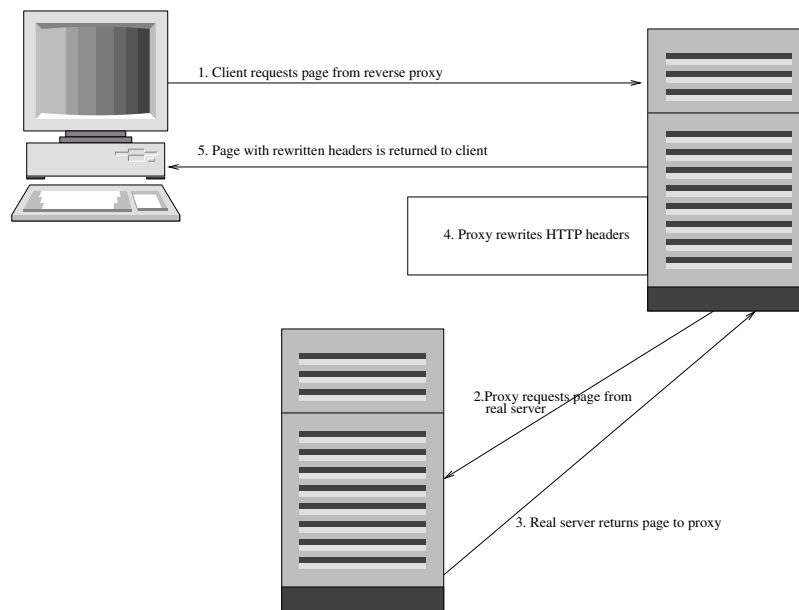


Figure 7: Reverse Proxying

```
<VirtualHost www.ripoffmicrosoft.com>  
    ProxyPass / http://www.microsoft.com/  
    ProxyPassReverse / http://www.microsoft.com/  
</VirtualHost>
```

Figure 8: Apache configuration for a reverse proxy

```
HTTP/1.1 404 Not Found
Date: Tue, 04 Feb 2003 20:45:48 GMT
Server: Apache/1.3.24 (Unix) PHP/4.2.1
X-Powered-By: PHP/4.2.1
Content-Type: text/html
```

Figure 9: Headers generated by the real server when a non-existent page is requested

```
HTTP/1.1 400 Bad Request
Date: Tue, 04 Feb 2003 20:48:26 GMT
Server: Apache/1.3.26 (Unix) PHP/4.0.6 mod_ssl/2.8.10 OpenSSL/0.9.6
Connection: close
Content-Type: text/html; charset=iso-8859-1
```

Figure 10: Headers generated by Apache Reverse Proxying host when a malformed HTTP/1.1 request is sent to it

Investigation of this form of masking presents a significant challenge and may, in fact, be impossible to detect in some cases. A fuller discussion of this is given in section 4

3.7 Forward proxy issues

In addition to reverse proxies, which perform header translation as described above, it is common for most networks to implement some form of forward proxy (sometimes also known as an HTTP accelerator).

Typically, forward proxies, are found at

1. the connection from the end-user to the public internet
2. at the entry-point to the target server network

A forward proxy acts as an intermediary on behalf of the end-user, receiving requests from their browser and fetching content from the target server. In the process of doing this it stores copies of retrieved content in its own cache so that it can return the cached copy in future. This leads to improved response time, when the same content is being retrieved regularly, while reducing bandwidth usage for the client and server networks.

In the past, use of forward proxies required manual configuration of the WWW browser, but many networks use the technique of “transparent proxying” [14] to force all users to use the proxy server in preference to a direct connection. This can frustrate attempts to investigate a rogue WWW server as all content is retrieved first from the proxy rather than the live server. Investigators should be aware of their local network configuration and must ensure that no forward proxies exist between them and the target network or, at least, that the content of the target server has not been cached prior to the start of the investigation.

Transparent proxying is also used at the server-end, to provide fast access to static content held on a server located behind the proxy. A cluster of transparent

proxies can provide an effective self-updating load-balancing pseudo-mirrored system. Generally, this mechanism should not provide any additional challenges for the investigator, except where the proxy is also capable of acting as a server. In this case, it should be treated as a server and examined as for any mirrored cluster environment (See section 3.2).

Forward proxies generally announce their presence by adding an HTTP “Via” header to all requests passing through them, but this behaviour is not guaranteed for all proxies, especially when running in transparent mode.

4 Detecting Reverse Proxies

4.1 Proxy signatures (HTTP Headers)

As mentioned previously, reverse proxies add no information to the HTTP headers, unlike forward proxies. Their effect is limited to translating absolute URL information, in the headers only, to refer the proxying server instead of the originating server. As a result, detecting a reverse proxy from a set of successful requests is, effectively, impossible.

Further, since the reverse proxy mapping is usually based on FQDN to allow use of HTTP/1.1 with its “Host” header to specify a VirtualHost within a server, it is possible for a single reverse proxy to map to multiple server instances if DNS round-robin clustering is in use on the originating (proxied) network.

Thus, the problems associated with clustered servers are now scaled. The front-facing public FQDN may be a clustered FQDN (of $c1$ hosts) implementing reverse proxies onto further clustered FQDNs (of at most $c2$ hosts), possibly using a non-public DNS system. As each reverse proxy may have a unique configuration this gives rise to worst-case situation where $c1 \times c2$ hosts or multiple independent servers may have to be located and examined.

Bearing in mind that the suspect content may be located on only one of proxied servers, this clearly makes the investigation a lengthy process, assuming the identities of the proxied hosts can be found.

4.1.1 Virtual Directories

The reverse proxying technique can be extended to provide “virtual directories” on a server, where individual directories, apparently on the file system, are mapped to different servers. Thus a single coherent WWW site using 20 directories, may in fact be fetching content, transparently, from 20 distinct servers or clusters around the world.

Directories may also be mapped to server side programs so that a URL of the form “http://www.cic.hull.ac.uk/dir/file.html” is actually triggering a server-side program rather than fetching a static file from a real directory. These are difficult to detect, but manipulation of the “file name” component of the URL may cause an error to be generated by the server-side program, thus disclosing its presence.

4.2 Approaches to investigation

To perform a rigorous investigation of any WWW site, given the complexities introduced above, we must first consider techniques for determining the nature

of the site being investigated. Once the site type has been identified, appropriate techniques can be selected to allow a comprehensive, fuller, examination to be undertaken.

4.2.1 DNS

The first point of contact should be the DNS records for the site under consideration. Use of the “nslookup” or “dig” commands to retrieve DNS entries for the FQDN will identify whether the site is using round-robin DNS or not. Furthermore, by employing a reverse lookup on the IP address, it will be possible to determine if the site in question is a real physical entity or a “virtual host” located on a server which hosts many sites.

If a single IP address is found to map to a single FQDN then the site in question may be amenable to the simple investigative techniques suggested at the start of this paper, but it is entirely possible that the machine identified is acting as a reverse proxy.

In this situation, extreme care must be taken as each and every file which is apparently present on the server may, in fact, be fetched from a remote server using reverse proxying.

If however, as is more likely, it is found that a number of IP addresses and host names are associated with each other, the more complex situation of DNS clustering and/or virtual hosting (which are regularly combined) must be considered.

4.2.2 HTTP variables and Cookies

It is possible that the suspect server may be changing its behaviour and supplying different content depending on the nature/identity of the client connecting to it [15]. A number of variables in the standard HTTP/1.x request header may be used including

- **REFERER** The URL of the previously visited WWW resource. Typically a page containing a link to the resource being requested.
- **USER_AGENT** A string identifying the client browser program. Many browsers allow this string to be customised.
- **ACCEPT_LANGUAGE** The preferred language of the user. Some servers can select different content files to deliver based on this.
- **Cookies**.

Cookies are small variables stored in the browser, at the request of a server, and volunteered back to the server each time a file is requested from it. Although these are notionally under the control of the server, it is possible to edit the content of cookies, thus a user can create an “artificial” cookie which has not originated on a server. The presence of this cookie may be used to gain access to hidden content by authorised users.

Thus cookies can be used in two ways :

- **User Tracking.** To track users as they navigate around the site and return to it after a period of time. A unique value will be set in each user's browser the first time they visit the site. This will subsequently uniquely identify them to the server (as a previous visitor, although it will not reveal any personal information unless the user has already disclosed it to the server)
- **User Authentication.** Because HTTP sessions are stateless, it is necessary to present a token to the server to prove that a user has correctly authenticated themselves at the start of a session. Cookies are commonly used as such a token. Indeed, the cookie itself may be the only authentication token required, as it can be constructed "by hand"

During examination of a site, the investigator should check the effects of denying and accepting cookies from a site. Equally, if they are investigating a site which has been identified by examination of data on a seized computer, they should give consideration to "borrowing" cookies relating to this site so that they appear to be an existing rather than new user. Care should be taken in the last case, as the combination of a known cookie with a new browser or IP address/range may be inappropriate. Similar consideration should be given to "custom" browser identification strings and other HTTP variables which can be used in the same way.

4.2.3 Network latency

It has been suggested that, because the reverse proxy is translating HTTP headers as information passes through it, that there may be some way to detect the presence of the proxy by measuring the delay between page request and final delivery. However, this is unlikely to provide much useful data as the translation occurs on all pages in the proxied server and hence there is no source of untranslated data for comparison. Furthermore, as the nature of the site in question (static HTML, dynamically generated HTML etc.) is inherently difficult to determine, an investigator who manages to identify a delay cannot be certain that this delay is caused by proxying rather than some other processing function occurring on the server, or delays inherent in the network itself.

A suggestion has been made that the use of traceroute, with its associated timing information, may be appropriate to detect the presence of a reverse proxy. The implication being that we might, somehow, be able to detect time differences between the proxying host and the real host. Brief consideration of this, however, highlights the simple fact that any traceroute must always be performed against the proxying host only as this is the only part of the system whose identity is known. Furthermore, even if we could generate a traceroute "through" the proxy to the proxied host, it is unlikely to be of much benefit as the proxied host is most likely to be protected by strong security.

4.2.4 Deliberate bad request

Rather than using traceroute, a more effective means of detecting whether a proxy is in use or not is the use of a deliberately "bad" page request, i.e. a request for a page which is known not to exist.

If the proxying host runs a different version of the server to the proxied host, then a request for an invalid URL or a syntactically badly-formed request (e.g HTTP/1.1 GET without the Host: field) is likely to generate a HTTP error from the proxy rather than the proxied host. The header information thus generated may disclose the identity or presence of the proxy. Similarly, a deliberately bad request for a page from within the proxied site will generate an error from the proxied host and may disclose further information about this server.

These requests can be generated simply through the use of a telnet program communicating directly with the server's HTTP port (usually port 80) (Fig. 11)

```
telnet testhost.cicnet 80

GET / HTTP/1.1
```

Figure 11: Use of telnet to generate badly-formed HTTP/1.1 GET request

If the proxying server receives a HTTP/1.1 request without the mandatory Host: field in the request header, then the proxy must generate a 400BadRequest response (Fig. 12). If HTTP/1.0 is used and the server is willing to handle HTTP/1.0 requests then it must return a default page.

```
HTTP/1.1 400 Bad Request
Date: Sun, 01 Dec 2002 21:18:47 GMT
Server: Apache/1.3.23 (Unix) (Red-Hat/Linux) mod_ssl/2.8.7
       OpenSSL/0.9.6b DAV/1.0.3 PHP/4.1.2 mod_perl/1.26
       mod_throttle/3.1.2
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html; charset=iso-8859-1

169
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<HTML><HEAD>
<TITLE>400 Bad Request</TITLE>
</HEAD><BODY>
<H1>Bad Request</H1>
Your browser sent a request that this server could not understand.<P>
client sent HTTP/1.1 request without hostname (see RFC2616 section 14.23): /<P>
<HR>
<ADDRESS>Apache/1.3.23 Server at proxytest.cicnet Port 80</ADDRESS>
</BODY></HTML>

0
```

Figure 12: Proxying server's response to badly-formed request. Note the presence of the proxying host's FQDN in the default HTML response page used.

4.2.5 Embedded information

If the use of deliberate “bad” requests does not prove successful, inspection of the content of the pages being returned may disclose useful information. This technique is possible because the proxying host does not rewrite HTML, only HTTP headers, thus the page content presented to the client is identical to that which is produced by the proxied host.

It is entirely possible that the creator of the original content has left some information about the proxied host, the tools they used or themselves within the HTML.

Fruitful areas for investigation include :

- **Embedded URLs**
Where the pages contains images, references to included files (e.g. stylesheets, scripts) or links to other pages, the author may have used absolute (i.e full) URLs in preference to relative URLs. These full URLs will contain the FQDN of the originating host for the file or link targeted. It may be possible to determine the identity of the proxied host by examination of these URLs.
- **Meta Tags**
Most pages contain additional information about the page to aid copyright enforcement and search-engine placement. Typical files include the author, generating tools, keywords and page description.
Such information is most likely to be found where the pages have been generated using content-management systems or WYSIWYG-style “authoring” tools such as Frontpage or Dreamweaver.
- **Visible footers**
Again, to aid copyright enforcement and search-engine placement and also usability (in accordance with Nielsen’s recommendations [16]) many page creators automatically include information at the bottom of the page. Typical fields include :
 - Copyright Notice
 - Last modified date
 - Creation date
 - Originating URL
 - Author or Owner

Careful inspection of the HTML source of the pages looking for both displayed information and non-displaying “META” and comment tags (beginning with “!-”) may prove fruitful. Some image files may also contain additional “watermarking” or descriptive information which will allow their origins to be traced.

5 The improved protocol - summary

In summary, therefore, the investigator should, before commencing examination of any WWW site, take steps, described above, to discover

- The full set of domains involved in the investigation. e.g. if the complain has arisen because of a WWW page, ALL sites referenced in the mail should be checked. Generally speaking, more information is better, so the full source of a WWW page or the complete text of an e-mail (including ALL headers) should be provided.
- Obfuscation. Special techniques may be required to deal with obfuscated HTML (i.e. the situation where apparent gibberish is transformed into a viewable web page through the use of a client-side “decryption” program) or other data. Even the most nonsensical text may be capable of becoming a useful source of information during an investigation.
- If DNS round-robin clustering is being used. DNS records must be checked before investigation begins to ensure that servers can be accessed by IP address instead of FQDN.
- Pre-browsing intelligence. If the investigation arises from recovery of data from a seized machine, the seized machine should be thoroughly examined for cookies, customised browser identification etc. which may be relevant to the site. It may be necessary to “impersonate” an existing user rather than become a new user on the site.’
- If URL masking using redirection or frames is in use. HTTP responses and HTML should be checked for the presence of 30x series responses or frameset code respectively.
- If proxying techniques are being employed to protect servers or directories. Servers should be probed using “invalid request” methods to attempt to generate 400 or 404 responses which may disclose the presence of an intermediate server. This should be done for ALL directories in referenced in the pages. All received HTML should be thoroughly checked for the presence of “signatures” indicating the real point of origin.
- HTML inspection. The full source of the “entry” pages should be thoroughly inspected to identify all obvious directories and files used. Non-viewable information should also be checked for.
- Directory probing. Directories referenced in the HTML should be probed thoroughly to check for the presence of reverse proxying or server-side programs mapped to them.
- Site behaviour. Behaviour with and without cookies should be checked.
- Non-obvious files. Attempts should be made to retrieve default directory listings, image directories, “robots.txt” and other files (e.g. “readme.txt”) which may exist on the server for use by software agents or non-casual users (i.e. associates of the criminals (ab)using the server). These files may contain reference to “hidden” content which is of interest / value to the investigator. Consideration should also be given the language in use in the country where the server is thought to be located. (e.g. for Germany a file called “lies.mich” is the equivalent of the USA/UK “read.me”)

Once the true nature of the server(s) hosting the site has been determined (as far as possible), the investigator should then proceed to produce accurate copies of the sites from each server involved. During the content examination, the investigator should also consider all meta-data which may further inform him/her about the nature of the hosting, clustering and/or proxying being used to protect or optimise the site in question. It may even be deemed appropriate to conduct “recursive” probing where proxying is suspected as there may be a chain of proxies in operation purely to confuse and/or delay the investigator.

Where the site is complex, containing many directories and a mixture of proxying and information-hiding techniques, care must be taken to ensure that the site owner is not alerted to the investigation. The use of different IP addresses, browsers and other tools is strongly recommended and investigators should not conduct too many error-generating probes at any one time.

Full examination of DNS records (both forwards and reverse) should be used as should examination of WHOIS records.

References

- [1] Marshall A.M. and Tompsett B.C. Spam 'n' Chips - a discussion of internet crime. *Science & Justice*. 2002; 42 : 117-122
- [2] Berners-Lee T. *Weaving the Web*. Texere 2000.
- [3] Hunt C. *TCP/IP Network Administration* 2nd edition. Sebastopol, CA, USA. O'Reilly, 1998: 338-341.
- [4] Davis C., Vixie P., Goodwin T. and Dickinson I. RFC876 - A Means for Expressing Location Information in the Domain Name System. <http://www.rfc.net/rfc1876.html>. 1996. Last viewed: 4th Feb. 2003
- [5] McAfee Security. McAfee Visual Trace. <http://www.mcafee.com/myapps/neoworx/default.asp> Last viewed: 4th Feb. 2003
- [6] *Data Protection Act 1998*. HMSO 1998.
- [7] Albitz P. and Liu C. *DNS and BIND* 4th edition. Sebastopol, CA, USA. O'Reilly 2001.
- [8] Rabinovich M and Spatscheck O. *Web Caching and Replication*. Addison-Wesley 2002
- [9] *The Captcha Project*. Carnegie-Mellon University. <http://www.captcha.net/> Last viewed: 4th Feb. 2003.
- [10] Koster M. A Standard for Robot Exclusion. <http://www.robotstxt.org/wc/norobots.html> 1994. Last viewed: 4th Feb. 2003
- [11] Niederst J. *Web Design in a Nutshell*. Sebastopol, CA, USA. O'Reilly 1999: 101.
- [12] Fielding R., Gettys J., Mogul J., Frystyk H., Masinter L., Leach P. and Berners-Lee T. RFC2616 HyperText Transfer Protocol - HTTP/1.1 World-Wide-Web Consortium 1999. <http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.htm> Last viewed : 4th Feb. 2003.
- [13] Apache Software Foundation. *Apache httpd*. <http://www.apache.org/> Last viewed: 4th Feb. 2003
- [14] Cohen A., Rangarajan S. and Singh N. Supporting Transparent Caching with Standard Proxy Caches. In: *Proceedings of the 4th International Web Caching Workshop*, March 1999
- [15] Spainhour S. and Eckstein R. *Webmaster in a Nutshell* 2nd Edition. Sebastopol, CA, USA. O'Reilly 1999 : 389-399 & 474-475.
- [16] Nielsen J. *Designing Web Usability: The Practice of Simplicity*. Indianapolis. New Riders Publishing 2002.