SmoothWall
delivering versatile, affordable security

Meeting the Challenges of Web Content Filtering

White Paper

SmoothWall, Meeting the Challenges of Web Content Filtering – White Paper, March 2007

Trademark notice

Acknowledgements

SmoothWall acknowledges the work, effort and talent of the SmoothWall GPL development team: Lawrence Manning, Richard Morrell, William Anderson, Dan Cuthbert, Dan Goscomb, Toni Kuokkanen, Jon Fautley, Pierre-Yves Paulus, Nick Woodruffe, Emma Bickley, Paul Tansom, Nigel Fenton and Adam Wilkinson.

Contact

SmoothWall Ltd.
1 John Charles Way
Leeds
LS12 6QA
United Kingdom

Tel: +44(0)870-1-999-500
Fax: +44(0)870-1-991-399
Email: info@smoothwall.net
Web: www.smoothwall.net

Contents

Introduction – what's the problem?

For many, access to the Internet is a mixed blessing; at worst, it can pose serious problems.

One of the primary concerns is productivity – not much will be accomplished in an ICT lesson if students are chatting on Meebo, prettifying their MySpace profiles and playing web-based games.

There are also moral and ethical obligations that parents will feel the school has, such as preventing children from accessing pornographic or racist websites from school computers. It is not realistic to expect teaching staff to supervise all computer use, especially if computers are accessible during breaks.

In corporate environments, tools are needed to enforce contractual obligations, such as a company's acceptable use policy for office computer equipment.

This paper discusses some of the problems associated with web content filtering, and poses potential solutions.

URL-based Filtering

Many programs block content classed as objectionable and market themselves as the solution to the web content filtering problem. However, most of these programs simply filter URLs, relying on a large database of pre-classified web addresses.

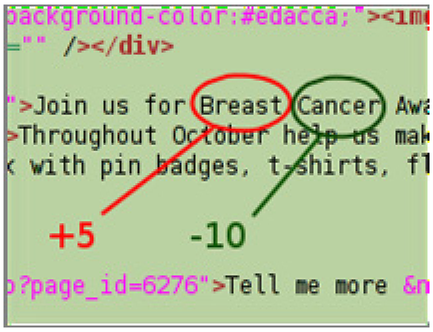
The major drawback with such software is that it can only effectively filter content that has already been screened by the maintainers of the database – if a site is not in the database, this is not necessarily because it is not undesirable; rather, the software is simply unable to make any useful classification.

Given that the Internet consists of tens of billions of pages with millions more added every day, such a system can realistically only cover a tiny percentage of existing sites, and will always be fighting a losing battle against creators of undesirable content. Some particularly naïve URL filters can also be very easily bypassed, as will be covered later.

What is SmoothWall web content filtering?

SmoothWall's web content filtering is a technically advanced method of filtering, which produces much better results, and is based on analyzing the actual content of a page. Using content analysis, SmoothWall Guardian tags particular words and phrases with a score and a category, and looks for occurrences of these in the source of web pages.

This enables SmoothWall Guardian to make meaningful decisions about content that has not previously been screened by human operators, something sorely missing from pure URL-based filters. Essentially, a page's content is scanned for any known words and phrases, the scores associated with each are summed, and the page blocked if the total score is above a configurable threshold value.



```
ackground-color:#edacca;"><im
="" /></div>

">Join us for Breast Cancer Awa
>Throughout October help us mak
< with pin badges, t-shirts, fl

?page_id=6276">Tell me more &n
```

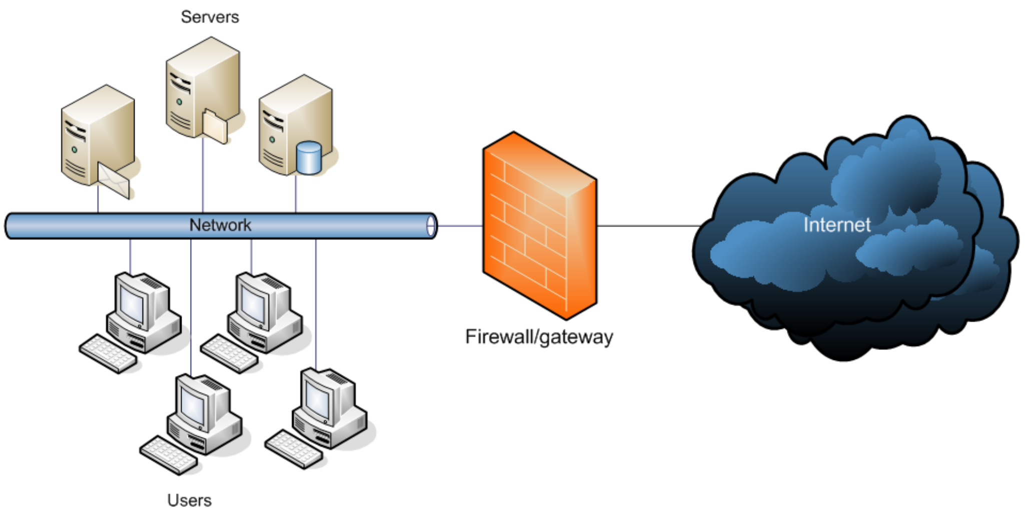
Flexibility and intelligence are provided by allowing phrases to have negative and positive scores. For example, a filter that banned pages containing the word 'breast' might effectively block some amount of pornography, but could also block a lot of medical material. On the other hand, a filter that assigned a negative score to ‘breast’ when found in combination with ‘cancer’ would stand a lesser chance of triggering such false positives.

However, SmoothWall Guardian’s content analysis gives you more than intelligent tagging. As the content of each and every web page requested is analyzed, malicious code, which can exploit HTML, JavaScript, Images, ActiveX, Java, is also identified and blocked.

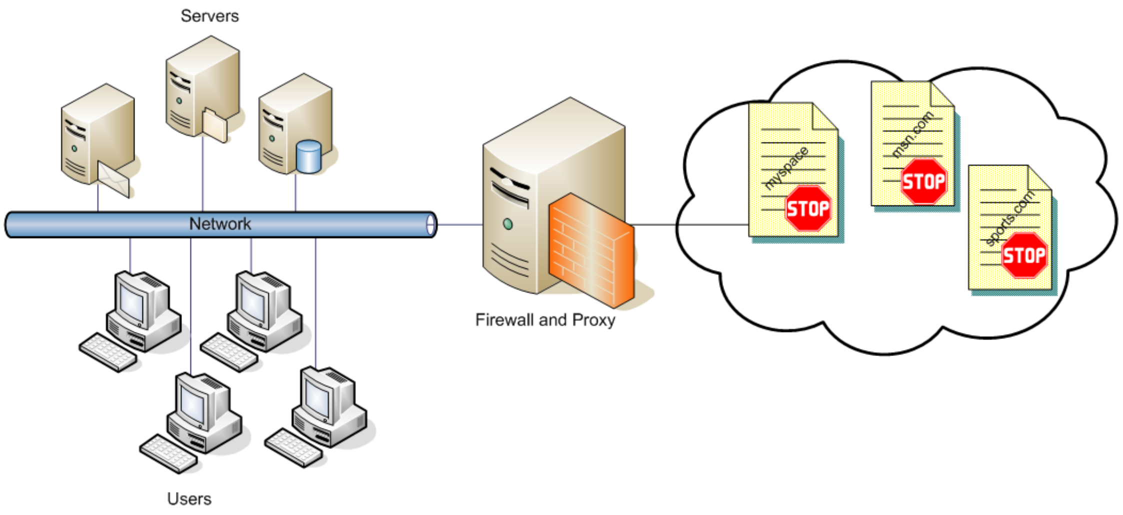
As well as providing dynamic analysis and protection against malicious code, SmoothWall Guardian also blocks search engines that return banned content in hits. Something no URL filter could ever do.

Terminology

Before going into more detail, let’s introduce a little terminology.



What we see above is a typical, simplistic network layout, consisting of client PCs, back-end servers and a firewall acting as the gateway to the Internet. By default, when a client PC requests a connection to the outside world, the firewall simply obliges – the connection is made and the traffic is forwarded, largely uninspected, straight to its destination.



In the graphic above, traffic is proxied. This means that rather than requesting connections directly to the outside world, client PCs request a connection to the gateway itself or to some other internal server, and send requests that effectively specify a 'forwarding address'.

The primary difference between forwarded and proxied traffic is that proxied traffic is reconstructed. Traffic is received by an application – such as the content filter – over which you have administrative control. This enables you to alter the traffic in any way, in both the outgoing and incoming directions, before being sent to its destination.

There is a third way in which traffic can make it to the outside world: "transparent" or "interception" proxying. In this case, traffic from clients destined for the Internet is silently redirected to an internal proxy server, without the browser's knowledge.

Transparent and Non-transparent Proxies

As mentioned above, proxying traffic comes in different flavors: transparent and non-transparent.

The advantage of transparent proxying is that it does not require any client-side configuration. Client PCs traffic is sent as normal, and the router intercepts and redirects it. However, it is not always reliable, as it is not always possible to retrieve the original destination address of a web request if it is not embedded in the request itself. Also, some methods for requiring users to authenticate before they can browse cannot be used, as a browser will not know how to respond to authentication requests from the proxy.

Secure HTTP (HTTPS) also presents a problem, as when a browser has not explicitly been told to use a proxy, traffic is encrypted – and so is largely unfilterable – as soon as it leaves the client.

Non-transparent proxying is much more flexible and reliable, although it does require that the web browsers on client machines be explicitly configured to send through the proxy. This is frequently viewed as a large drawback, perceived as hard to deploy and/or easy for end users to modify, but these are both myths on properly managed networks.

Deploying and Protecting Proxy Settings

One way to set up PC clients for manual proxying would be to simply go around to each client machine individually and enter the proxy settings. Not only is this incredibly tedious, but it is also a very bad idea, as users will simply be able to open up their browser's preferences and change them.

A much better idea is to have browser settings locked down, with the locked down values themselves centrally administered.

In the case of Internet Explorer, this is accomplished using Group Policies, which is very well integrated with standard Windows network administration as one would expect.

Opera comes second in the ease-of-administration stakes, with the concept of the 'system fixed file'. Options users should not be able to change can be written to this file, in the same format as the default and user-specified options. This file can then be pushed to clients as, for example, part of a login script.

Firefox makes life a little more difficult in this regard, but it is still possible – a link to one suggested procedure can be found at the end of this document. Two projects also exist to provide Administrative Templates for Firefox settings, allowing administration through Group Policies.

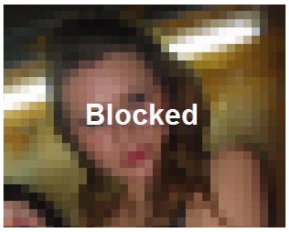
Bypassing Web Content Filters

There are two basic ways to bypassing web content filters: getting around them, meaning that traffic is not passed through the filter; and getting through them, where traffic is passed through the filter, but the address of content – and/or the content itself – is obscured in some way.

Getting *Around* Web Content Filters

Unfortunately, users who do not have permission to install new software, or to change the proxy settings in existing browsers, still have options open for getting their traffic around filters. There are browsers which do not require installation, and which will ignore locked-down settings. Portable Firefox, for example, can run entirely self-contained from a folder on a USB pen-drive. Users can then simply browse without any proxy.

You can block forwarded traffic on ports 80 and 443 to prevent this, but really determined users will find or set up their own proxies, outside the local network, accepting traffic on other ports, and browse using these.



To counter this sort of activity, you could implement a default-deny policy on your firewall, only allowing traffic to be forwarded on specific ports. However, this alone does not put any limits on the nature of traffic on these ports, meaning they can still be used for talking to proxies.

Explicitly Proxying

There are two ways you can stop the use of external proxies: implement a default-deny policy in conjunction with Intrusion Detection System (IDS) software, to limit usable ports and monitor the type of traffic flowing over them; or simply refuse to forward traffic.

The former requires maintaining complex rules to identify the protocols one expects to see in use, and incurs a performance overhead on the gateway. The latter simply means setting a default-deny policy with no exceptions other than for known, internal proxy servers. Run proxy servers – filtering or not – for protocols you wish to make available to clients, and deny everything else.

In fact, client PCs can function perfectly well without even having a default gateway configured. The only traffic that can make it to the outside world is proxied traffic: proxies exist for many protocols besides HTTP, and by enforcing their usage, you are forcing people not only to use expected ports, but expected protocols.

Generally speaking, if a proxy server does not exist for a given protocol, it is not a protocol you want your clients to be using. You also gain logging, and – for supporting protocols – authentication, for everything going on between non-privileged clients and the Internet.

Getting *Through* Web Filters

Once users have been prevented from getting round the web content filter, the remaining concern is that it is able to determine what content is actually passing through it.

It is in this regard that URL-only filters really show their limitations. Some systems that work this way are easily bypassed by simply looking up a website's IP address, and using this for accessing the site rather than the domain name.

For example, 83.138.146.86 resolves to www.smoothwall.net. Entering http://83.138.146.86/ into the browser's address bar may bypass a naïve filter which looked statically for www.smoothwall.net in the address.

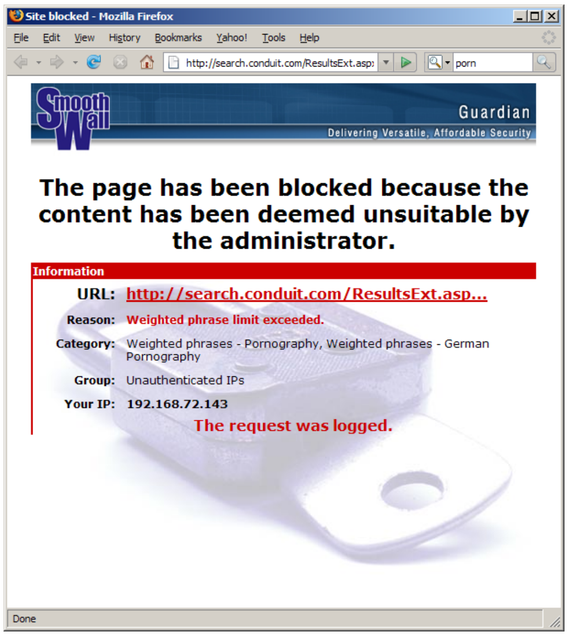
Other URL-obscuring tricks include looking at the Google cache of a website, which effectively hides undesirable content behind a domain name that is unlikely to be blocked, and using web-based proxies.

Web-based proxies are web pages containing a form into which URLs can be entered, typically returning the requested content in a frame, appearing to the web content filter to be part of the site hosting the form. Such proxies are very simple to install, making it effectively impossible for a static URL database to catch all of them. If a web-based proxy is hosted on an HTTPS site, then web content filtering becomes even more difficult, as even the content is obscured from the filter.

However, SmoothWall does not totally dismiss URL filtering as it is a good way to block non-work related content such as news, sport, shopping and travel and we use categorized URL, domain and IP address blocklists as secondary filtering mechanisms. But remember, URL filtering is useless for things like child pornography sites where the domain will only be used for a couple of days and the baddies move in order to keep one jump ahead of the authorities.

Catching Web-based Proxies

A true web content filter like SmoothWall Guardian is immune to most URL obscuring tricks. Any site blocked based on textual content is still caught when looked at using the IP address directly, or through Google's cache, or similar, since the content of the page itself is unchanged.


Site blocked - Mozilla Firefox
File Edit View History Bookmarks Yahoo! Tools Help
http://search.conduit.com/ResultsExt.asp:
porn
Guardian
Delivering Versatile, Affordable Security

The page has been blocked because the content has been deemed unsuitable by the administrator.

Information

URL: http://search.conduit.com/ResultsExt.asp...
Reason: Weighted phrase limit exceeded.
Category: Weighted phrases - Pornography, Weighted phrases - German Pornography
Group: Unauthenticated IPs
Your IP: 192.168.72.143

The request was logged.

Done

SmoothWall Guardian also has a Deep URL scanning option, which essentially looks for URLs within URLs.Some search engines, such as Google Images, will embed the original source address of content into the address used to access cached or thumb-nailed versions. SmoothWall Guardian can detect this, and can check its database for both addresses. This also applies to some web-based proxies. Although limited to entries in the URL database, this is still a powerful feature.

SmoothWall Guardian's blocklists contain the characteristics of web-based proxies, and enable SmoothWall Guardian to identify and block the actual forms before they can begin being used for browsing. The vast majority of web-based proxies are unmodified installations of a handful of pieces of software, each with their own identifying characteristics – button names, option strings, etc. – which are usually unmodified and detectable.

As noted earlier, web-based proxies hosted on HTTPS sites present a particular challenge. If using transparent proxying, there is nothing at the gateway level that can be done to detect their usage, as the encryption is end-to-end. However, if clients have been configured to use a proxy server, then the door is opened for filtering by destination domain, as the initial request for an encrypted tunnel through the proxy is made in the clear.

Couple this with the fact that HTTPS is typically only provided where necessary – for example, it is traditionally used by banks and shops, but not news or research sites – and it becomes feasible to impose a blanket block on HTTPS access, maintaining a small whitelist of accessible sites, without being viewed as overly restrictive.

SmoothWall Guardian lends itself well to implementing this kind of configuration, with blanket blocking options for HTTP and/or HTTPS covering either all requests or requests for bare IP addresses, the latter commonly indicating untrusted sites or attempts to bypass filtering.

Torpark – the miscreant's best friend?

There is an emerging trend towards incorporating privacy and encryption functionality into any and all communications software, including web browsers. One of the recent developments along these lines is Torpark, which combines the zero-installation Portable Firefox with Tor, 'The Onion Router', an encrypted communications framework aimed at re-establishing and protecting the anonymity that Internet users used to enjoy.

Traffic is encrypted from the client through to the Tor circuit endpoint, having been routed through an unspecified number of intermediate nodes, and it is not guaranteed that either the route it takes or the ports it uses will be in any way static. Assuming that the project has met its design goals, there is no simple way to determine the original protocol of traffic encapsulated in the Tor network, and certainly no way to ascertain the actual content being transported within.

However, if explicit proxying is in use, the software simply does not function. Even though it can be configured to use an HTTP proxy, experimentation has shown that only the initial circuit negotiation will be performed via the proxy.

At the time of writing, the actual encrypted traffic is simply expected to be forwarded to its destination by the gateway. The HTTP requests made during proxied circuit negotiation are also quite distinctive: pure IP addresses rather than domain names; very large filenames with a constant length, extension and MIME type. It would not be infeasible to create regular expressions within SmoothWall Guardian to block such requests.

If explicit proxying is too restrictive for your network, but you still do not like the idea of Tor clients or external proxies being used, then as hinted at earlier, IDS software may come in useful.

The 'Bleeding Snort' archives contain a wide array of rules for the Snort IDS, compatible with the SmoothIDS; these include detection of Tor circuit traffic, HTTP requests on non-standard ports, and SSL handshakes on non-standard reports.

In Conclusion

At the end of the day, locking down a network is a trade-off between flexibility and control.

Wherever you choose to draw the line, it is important to be aware of the gray areas in your network traffic, the general level of technical expertise amongst your users, and how life can be made more difficult for those seeking to undermine acceptable usage policies.

References

http://ilias.ca/blog/2005/03/locking-mozilla-firefox-settings/ – Unofficial how-to for locking Firefox options

http://wetdog.sourceforge.net/ – The WetDog project provides administrative templates and GPO integration for Firefox 1 and 2

http://fxcorp.sanduskycomputers.com/ – The Firefox Corporate project, another GPO intergration project, also provides extra deployment and management functions

http://tor.eff.org/ – Tor, The Onion Router and http://www.torrify.com/ – Torpark: Portable Firefox and Tor

http://www.bleedingthreats.net/ – Bleeding Edge Threats, home of the Bleeding Snort rule collection