# Freedom 2.0 Ad Manager

George Showman – georgesh@zeroknowledge.com
Joshua Svatek – joshua@zeroknowledge.com
Paul Branchaud – paul@zeroknowledge.com
Zero-Knowledge Systems, Inc.

February 7, 2001

***Abstract***

This white paper, targeted at the technically knowledgeable reader, offers a detailed look at the Freedom® software's Ad Management feature. It is intended to give the reader a good understanding of how web ads are handled when using the Freedom software.

## Introduction

As the web becomes more crowded, it is becoming increasingly difficult to visit a site that does not display ads in one form or another. Ads are a way for the web page's owner to generate revenue for the site. However, from a web surfer's perspective, ads can compromise individual privacy and detract from the actual content of the site, making navigation more difficult.

Version 2.0 of the Freedom software from Zero-Knowledge Systems adds a new feature that gives individual web surfers control over which ads (if any) are displayed in their browsers. There are a number of reasons why web surfers would want to turn ads on or off; Freedom allows users to make the choice themselves, rather than be forced to accept unsolicited advertisements.

## Ad Manager Overview

The World Wide Web functions on a request-response model: a browser makes an HTTP request for a web page from a server and the server returns an HTTP response, containing the relevant data. The Freedom software already monitors and processes all HTTP traffic; therefore, inclusion of the Ad Manager feature was a natural extension of the product's functionality.

When loading a web page that contains text and graphics, there are multiple request-response exchanges to transfer all the text and images. Most web ads are linked image files that require that the browser make a separate HTTP request-response exchange with the ad server (often a server that is not associated with the web page in question). Often, the browser's request to the ad server contains previously saved cookie information that allows the server to recognize the user, serving an ad that it believes to be relevant to that user's interests.

In order to manage web ads, Freedom blocks all HTTP requests to ad servers. Therefore, the ad server never receives the browser's requests (or any cookie information). Blocking web ads requires that the Freedom software monitor the HTTP data streams for requests to ad servers. Allowing users to enable Freedom's Ad Manager improves the surfing experience by reducing a page's download time and preventing cookie-based tracking.

## Ad Manager Benefits

Because Freedom prevents the browser from sending HTTP requests to known ad servers, the user does not need to wait to download data that is unrelated to the page's actual content. There are two primary benefits to using the Freedom software's Ad Manager feature:

- **Eliminates Cookie-Based Tracking:** This benefit may seem somewhat confusing, especially when considering that the Freedom software offers per-nym Cookie Jars® to address tracking issues. Furthermore, it is presumably acceptable for individual nyms to be tracked around the Internet, as they are separate from the user's real-world identity. However, the use of cookies to track individuals by ad servers (e.g.: doubleclick.net) adds little, if any, value to the browsing experience. In addition, such tracking can be considered a significant infringement of privacy. Ads blocked by Freedom never have the chance to pass identifiable information (cookies) to the ad server.

- **Improves Download Speed:** Many popular web pages contain few images related to their content; most images are advertisements. Often, the size in bytes of the ad images is larger than the size of the page's useful content (several hundred lines of text contain less data than a single ad-sized image). By intercepting and blocking a web browser's ad requests, Freedom's Ad Manager reduces the total number of bytes that are downloaded, saving time for users who are interested in content rather than ads. This bandwidth savings is in addition to the time saved by not having to read or watch the ads.

## Implementing the Ad Manager

While implementing the Ad Manager feature, several technical considerations had to be overcome to maintain the Freedom software's seamless transparency to the user. The first such consideration is that the web page's HTML code contains a link to the ad server with an image link. As the browser processes the HTML code, it sends a request to the ad server for the image. Simply blocking the request for the image would result in the browser waiting for a response and eventually timing out (as if the ad server were offline). The browser would eventually replace the missing image with a placeholder, indicating where the image should appear. Often, the browser's timeout value exceeds the time required to download the image file. Clearly, this result would not benefit the web surfer.

Although Freedom cannot account for server downtime, the user should be able to differentiate between a failed HTTP request and a blocked ad. Because the browser sent a request for the image, Freedom serves a "dummy" ad as the response to the original request. The dummy ad is a properly formatted HTTP response that contains a small, transparent image (a .GIF file). The browser accepts this image as the ad it originally requested, inserting it in the ad's place. The result is a blank space for each ad that is replaced on the page. Each dummy ad can still be clicked to link the user to the advertiser.

Many ad servers base their payments on the number of "click throughs" that a website generates. Because the Ad Manager prevents the HTTP request from being passed to the ad server, the cookie that identifies a user as having clicked a web ad is never placed on the user's machine.

### Identifying Ad Server Requests

There is no simple way to identify an HTTP request as a request for an advertisement. Therefore, the Ad Manager must maintain a list of known ad servers, and compare every HTTP request it intercepts with this list. This "block list" contains URLs or sub-strings of URLs for most of the popular ad servers. The Ad Manager bases its decision on whether or not to block an HTTP request by checking if the requested URL (e.g.: http://www.adserver.net/ads/can/), or some sub-string of it, is contained in the list. If the URL is included in the block list, Freedom discards the request and responds with the previously described dummy ad. If the URL is not included in the block list, Freedom sends the request on to its destination web server (subject to other processing by the Freedom software not related to ads).

The block list is a file named `adblocker.dat`, stored in plain text on the user's machine. As new advertising sites constantly appear, and existing sites change their addresses, the block list must be updated frequently to remain effective. Therefore, each time the Freedom software starts it requests a new version of the block list from a central, Zero-Knowledge®-maintained script server. This script server is described in relation to other Zero-Knowledge servers in the “Freedom 2.0 System Architecture” document[1].

This update is a simple HTTP request and response transaction, where the block list update arrives as a signed section in the HTTP response. The script server is updated periodically by the Freedom Support team. Details on how the block list is formatted and its syntax are described below, in the Block List Details section.

## User Control

Users of the Freedom software can customize the Ad Manager to suit their individual preferences. They can choose whether or not to use the Ad Manager by clicking or clearing a single check box. Individual users may want to modify their block lists by adding or removing specific ad servers. This gives each user total control of which ads they see.

For example, if the default block list contains ad servers that a user would like to receive ads from, they can remove these servers from the list. Furthermore, if a user receives ads from server that is not included in the block list, they can create new list entries using the URLs of ad servers not already in the list.

## Block List Details

As previously stated, the block list is kept locally on the user’s machine. The block list is a plain text file, and its syntax is quite simple. The following example is typical of a block list’s entries:

```
*~AdotherAT&T/555-1212.com<http://www.555-1212.com/images/att_iresban-lrg.gif
*~AdBargainDog<http://images.about.com/specials/service/promotion/sweepstakes/bargaindog.gif
~LA Times<www.latimes.com/RealMedia/ads/
~About.com<images.about.com/specials
~Adfishseries<www.worldbillfishseries.com/RealMedia/ads
@Adtrader Website Exception<www.adtrader.com
~Doubleclick<doubleclick.net
~The Onion<adbot.theonion.com
```

### Block List Syntax

The first character for each entry is a processing flag. The flag is either a tilde (~), an asterisk (*), or an "at" sign (@). The following table explains each flag's meaning:

| Processing Flag | Block List Syntax |
|---|---|
| **~** | Indicates that the listed ad server should be blocked. |
| ***** | Indicates an ad server that has been added manually by the user (through the Ad Manager dialog). |
| **@** | Indicates exceptions—URLs that specifically should **not** be blocked. |

After the processing flag, each line in the block list has an optional natural language name. The natural language name is followed by a separator (**<**) and then the URL or (sub-string) that the Ad Manager compares against every outgoing HTTP request.

#### Exceptions

In the previous example, requests matching any URL with `adtrader.com` as a sub-string will be blocked **unless** `www.adtrader.com` (the exception) also matches the HTTP request's URL. This exception allows a user to browse adtrader.com's main website while blocking all ads served from adtrader.com.

### Block List Updates

Updates from the script server are simply new block lists, consisting of new entries that the Ad Manager appends to the end of the locally stored list. The script server maintains a unique identifier for each line, so the Ad Manager only requests the correct number of entries by recalling the last line number it received from the previous update.

The previously described "asterisk" (*) processing flag allows the Freedom software to differentiate between manually added entries and updates received from the script server. The script server can delete entries from the block list by prefixing a line in the update with an exclamation point (!), but the Ad Manager will never remove a user-defined entry from the list. It should be noted that a user will never see an exclamation point entry in the `adblocker.dat` file, because the Ad Manager processes those lines (deletes the entries) before saving new data to the file.

## Conclusion

The Ad Manager is designed to help web surfers make the most of their time online. Web ads can be compared to roadside billboards; however, billboards do not affect a car's speed or the time it takes to reach a destination. The increased presence of web ads directly affects how quickly web surfers can access the information they seek. The Freedom Ad Manager gives users the choice to view ads or to turn them off. Control over the speed at which a web page is displayed is now in the hands of the user.

## Notices

Freedom, Zero-Knowledge, and Cookie Jars are trademarks of Zero-Knowledge Systems Inc. These trademarks are registered in certain jurisdictions.

## Endnotes

[1] “Freedom 2.0 System Architecture”, Philippe Boucher, Adam Shostack, Ian Goldberg, Zero-Knowledge Systems, Inc. http://www.freedom.net/info/whitepapers/Freedom_System_2_Architecture.pdf