Network Products

Traffic Server 4.0

Administrator’s Guide

Document Release 4.0
October 13, 2000

Inktomi®

Contents

List of Procedures

Preface

This manual describes how to use and configure an Inktomi Traffic Server™ system.

For information about installing Traffic Server, refer to the *Traffic Server Installation Guide*. For a information about unsupported features and last minute information not available in this manual, refer to the *Release Notes*.

The manual discusses the following topics:

- *Chapter 1* provides an overview of the Traffic Server features and components
- *Chapter 2* through *Chapter 12* provide procedural information about starting, monitoring, configuring, and maintaining the Traffic Server
- *Appendix A* through *Appendix F* provide Traffic Server reference information

Who should read this manual

This manual is intended for Traffic Server system administrators who configure, run, and administer Traffic Server systems.

The manual assumes that you have experience in UNIX or Windows and Web server administration, and that you are comfortable performing complex system configuration tasks, such as partitioning and formatting disks, setting up TCP/IP ports, and establishing DNS round robin services.

Conventions used in this manual

This manual uses the following typographic conventions.

Convention	Purpose
italics	Represent emphasis and introduce terms; for example, “the *reverse proxy* option.”
bold	Represents graphical user interface options and menu names; for example, click the **Protocols** button.
`monospaced face`	Represents commands, file names, file content, and computer input and output; for example, “use the `reconfigure` command.”
`monospaced italic`	Represents variables for which you should substitute a value; for example, “enter *`filename`*.”
brackets [ ]	Enclose optional command arguments in command syntax; for example, `add` *`pathname`* `[`*`size`*`]`.
vertical bar |	Separates value options in command syntax; for example, `open tcp`|`udp ports` *`o_ports`*.

Chapter 1

Overview

Welcome to a faster network.

Traffic Server speeds Internet access, enhances web site performance, and delivers unprecedented web hosting capabilities.

This chapter discusses the following topics:

What is Traffic Server?

The dream of global data networking has come true. Internet users request billions of documents each day all over the world. Unfortunately, this dream of global data networking has become a nightmare for information systems professionals as they struggle with overloaded servers and congested networks, trying to keep pace with society's growing data demands.

Traffic Server is a high-performance web proxy cache that improves network efficiency and performance by caching frequently accessed information at the edge of the network. This brings content physically closer to end users for faster delivery and dramatically reduces bandwidth usage.

Traffic Server is designed to improve content delivery for enterprises, Internet Service Providers (ISPs), backbone providers, and large intranets by maximizing existing bandwidth.

Traffic Server deployment options

Traffic Server can be deployed in different ways to best suit your needs and your environment:

✔ As a web proxy cache

✔ As a reverse proxy

✔ In a cache hierarchy

✔ In a Traffic Server cluster

The following sections provide a summary of the Traffic Server deployment options.

Traffic Server as a web proxy cache

As a *web proxy cache*, user requests for web content go to Traffic Server on the way to the destined web server (origin server). If Traffic Server contains the requested content, it serves it directly. If Traffic Server does not have the requested content, Traffic Server acts as a proxy, fetching the content from the origin server on the user's behalf, while keeping a copy to satisfy future requests.

Traffic Server provides two proxy caching options:

✔ *Transparent proxy caching*, where user requests are automatically injected into a Traffic Server cache on their way to the eventual destination. Users request Internet content as usual without any browser configuration and Traffic Server automatically serves their requests. The user's client software (typically a browser) is unaware that it is communicating with Traffic Server. Transparent proxy caching is described in more detail in *Chapter 3, Web Proxy Caching*.

✔ *Explicit proxy caching*, where the user's client software must be configured to send requests directly to the Traffic Server.

Traffic Server as a reverse proxy

As a *reverse proxy*, Traffic Server *is* configured to be *the* origin server the user is trying to connect to (the origin server's advertised host name resolves to Traffic Server, which is acting as the real origin server). The *reverse proxy* feature is also called *server acceleration*. Reverse proxy is described in more detail in *Chapter 5, Reverse Proxy and HTTP Redirects*.

Traffic Server in a cache hierarchy

Traffic Server can participate in flexible *cache hierarchies*, where Internet requests not fulfilled in one cache can be routed to other regional caches, taking advantage of the contents and proximity of nearby caches. In a hierarchy of proxy servers, Traffic Server can act either as a parent or child cache, either to other Traffic Servers or to other caching products.

Traffic Server supports the standard Internet Cache Protocol (ICP) to interoperate with existing ICP cache hierarchies.

Hierarchical caching is described in more detail in *Chapter 7, Hierarchical Caching*.

Traffic Server in a cluster

Traffic Server scales from a single node into multiple nodes that form a *cluster* allowing you to improve system performance and reliability. Traffic Server detects the addition or removal of nodes automatically. If Traffic Server's *virtual IP failover* option is enabled, Traffic Server maintains a pool of virtual IP addresses that it assigns to the nodes of the cluster. Traffic Server can detect hard node failures (such as power supply or CPU failures) and reassign IP addresses of the failed node to the remaining operational nodes automatically.

Traffic Server has two clustering modes:

- ✔ *Management-only* mode, where you can administer all the nodes in a cluster at the same time. Nodes automatically share configuration information.
- ✔ *Full-clustering* mode, where the node caches act as a single aggregate cache. A Traffic Server cluster distributes its cache across its nodes into a single, virtual object store, rather than replicating the cache node by node.

  A fully clustered Traffic Server provides a single system image to both users and administrators, appearing as a single virtual server. Full-clustering mode includes management-only mode.

Traffic Server clusters are described in more detail in *Chapter 6, Traffic Server Clusters*.

Traffic Server components

Traffic Server consists of several components that work together to form a web proxy cache you can easily monitor and configure. The main components are described below.

The Traffic Server cache

The *Traffic Server cache* consists of a high speed object database called the *object store*. The object store indexes objects according to URLs and associated headers. Using sophisticated object management, the object store can cache alternate versions of the same object, varying on spoken language or browser type, and can efficiently store very small and very large documents, minimizing wasted space. Once the cache begins to fill, the Traffic Server mobilizes garbage collectors to remove stale data, ensuring that the most requested objects are kept on-hand and fresh.

Traffic Server is designed to tolerate total disk failures on any of the cache disks. If the disk fails completely, Traffic Server marks the entire disk as corrupt and continues using the remaining disks. If all of the cache disks fail, Traffic Server goes into proxy-only mode.

You can partition the cache to reserve a certain amount of disk space for storing data for specific protocols and origin servers.

The Traffic Server cache is described in more detail in *Chapter 8, Configuring the cache*.

RAM cache

Traffic Server maintains a small RAM memory cache of extremely popular objects. This *RAM cache* serves the most popular objects as fast as possible and reduces load on disks, especially during temporary traffic peaks. You can configure the RAM cache size to suit your needs (refer to *Changing the size of the RAM cache, on page 108*).

The Adaptive Redirection Module (ARM)

The Adaptive Redirection Module (ARM) is used in *transparent proxy caching* to redirect intercepted user requests destined for an origin server to the Traffic Server. Before the traffic is redirected by the ARM, it is intercepted by an L4 switch or router.

To redirect user requests to Traffic Server, the ARM changes an incoming packet's address. The packet's destination IP address is changed to the IP address of Traffic Server and the packet's destination port is changed according to the protocol used. For example, for HTTP, the packet's destination port is changed to Traffic Server's HTTP port (usually 8080).

The ARM supports automatic bypass of sites that do not function properly with proxy caches.

Traffic Server can respond to client request overloads by forwarding requests directly to origin servers. This feature is called *load shedding*. Overload conditions, such as network outages, misconfigured routers, or security attacks, can slow down Traffic Server's response time. In transparent configurations, Traffic Server can use its ARM bypass functionality to forward overload requests directly to origin servers, bypassing the cache. When the overload condition dissipates, Traffic Server automatically returns to full caching mode.

The Host Database

The Traffic Server host database stores the Domain Name Server (DNS) entries of origin servers to which Traffic Server connects to fulfill user requests. This information is used to adapt future protocol interactions to optimize performance.

Among other information, the host database tracks:

✔ DNS information (for fast conversion of host names to IP addresses)

✔ The HTTP version of each host (so advanced protocol features can be used with hosts running modern servers)

✔ Host reliability and availability information (to avoid making the user wait for non-functional servers)

The DNS Resolver

Traffic Server includes a fast, asynchronous DNS resolver to streamline conversion of host names to IP addresses. Traffic Server implements the DNS resolver natively, directly issuing DNS command packets, rather than relying on slower, conventional resolver libraries. Many DNS queries can be issued in parallel and a fast DNS cache maintains popular bindings in memory, significantly reducing DNS traffic.

Traffic Server processes

Traffic Server contains three processes that work together to process Traffic Server requests and manage, control, and monitor the health of the Traffic Server. The three processes are described below:

- ✔ The `traffic_server` process is the transaction processing engine of Traffic Server. It is responsible for accepting connections, processing protocol requests, and serving documents from the cache or origin server.
- ✔ The `traffic_manager` process is the command and control facility of the Traffic Server, responsible for launching, monitoring, and reconfiguring the `traffic_server` process. The `traffic_manager` process is also responsible for the Traffic Manager UI, the proxy auto configuration port, the statistics interface, cluster administration, and virtual IP failover.

  If the `traffic_manager` process detects a `traffic_server` process failure, it instantly restarts the process but also maintains a connection queue of all incoming requests. All incoming connections that arrive in the several seconds before full server restart are saved in the connection queue and processed in first-come, first-served order. This connection queueing shields users from any server restart downtime.
- ✔ The `traffic_cop` process monitors the health of both the `traffic_server` and `traffic_manager` processes. The `traffic_cop` process periodically (several times each minute) queries the `traffic_server` and `traffic_manager` process by issuing heartbeat requests to fetch synthetic web pages. In the event of failure (if no response is received within a time-out interval or if an incorrect response is received), `traffic_cop` restarts the `traffic_manager` and `traffic_server` processes.

Figure 1 illustrates the three Traffic Server processes.

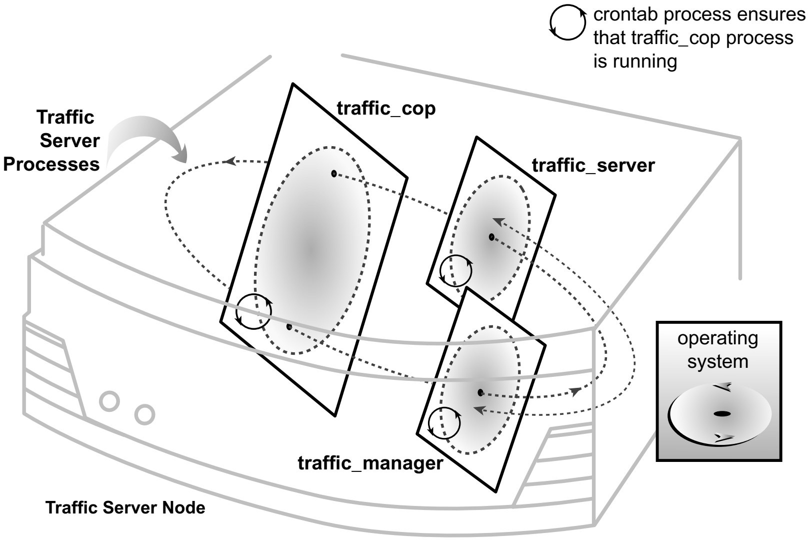


Figure 1 Traffic Server processes

Administration tools

Traffic Server offers several administration alternatives to suit the needs of many environments:

- ✔ The *Traffic Manager* User Interface (UI) is a web based interface consisting of a series of web pages accessible through a browser. The Traffic Manager UI provides a rich set of graphs and statistical displays for monitoring Traffic Server performance and network traffic, and a set of options for configuring and fine-tuning the Traffic Server system. The Traffic Manager UI offers password-protected, SSL-encrypted, single-point administration for an entire Traffic Server cluster.
- ✔ The *Traffic Line* command-line interface is a text based interface that provides equivalent functionality to that of the Traffic Manager UI. From the command line, you can execute individual commands or script a series of commands in a shell.
- ✔ Various *Configuration files* allow complete administration through a simple file editing and signal handling interface. You can change configuration options by editing configuration files manually instead of using the Traffic Manager UI or Traffic Line. (Any changes you make through the Traffic Manager UI or Traffic Line are automatically made to the configuration files.)

Traffic analysis options

Traffic Server provides several options for network traffic analysis and monitoring:

- ✔ *Traffic Manager statistics and graphs* show network traffic information. You can view graphs and statistics from the Traffic Manager UI or collect and process statistics using the command-line interface, Traffic Line.
- ✔ *MRTG* (Multi Router Traffic Grapher) is a graphing tool that provides a variety of graphs showing historical information about virtual memory usage, client connections, document hit rates, and so on. You can access MRTG from the Traffic Manager UI.
- ✔ *SNMP Network Management* support lets you monitor and manage Traffic Server through SNMP network management facilities. Traffic Server supports two management information bases (MIBs): MIB-2, a well known standard MIB, and the Inktomi proprietary Traffic Server MIB that provides more specific node and cluster information.
- ✔ *Traffic Manager alarms* are presented in the Traffic Manager UI. Traffic Server signals an alarm for any detected failure condition. You can configure Traffic Server to send E-mail or page support personnel when an alarm occurs.
- ✔ *Transaction logging* lets you record information in a log file about every request that Traffic Server receives and every error it detects. By analyzing the log files, you can determine how many people use the Traffic Server cache, how much information each person requested, and what pages are most popular. You can also see why a particular transaction was in error and what state the Traffic Server was in at a particular time. For example, you can see that Traffic Server was restarted or that cluster communication timed out.

  Traffic Server supports several standard log file formats, such as Squid and Netscape, and its own custom formats. You can analyze the standard format log files with off-the-shelf analysis packages. To help with log file analysis, you can separate log files so that they contain information specific to protocol or hosts.

Traffic analysis options are described in more detail in *Chapter 9, Monitoring Traffic*. Traffic Server logging options are described in *Chapter 12, Working with Log Files*.

Traffic Server security options

Traffic Server provides numerous options that enable you to establish secure communication between the Traffic Server system and other computers on the network. Using the security options, you can:

- ✔ Control client access to the Traffic Server proxy cache.
- ✔ Control which hosts are allowed to access the Traffic Server machine (ARM security).
- ✔ Configure Traffic Server integration into your firewall and control traffic through a SOCKS server.
- ✔ Configure Traffic Server to use multiple DNS servers to match your site's security configuration. For example, you might choose to have Traffic Server use different DNS servers depending on whether it needs to resolve host names located inside or outside a firewall. This enables you to keep your internal network configuration secure while continuing to provide transparent access to external sites on the Internet.
- ✔ Use LDAP-based proxy authentication that enables you to leverage existing directory services by supporting asynchronous match and bind requests to LDAP servers. This enables you to maintain policies that require users to log in and be authenticated by the proxy before going out onto the Internet. In addition, you can enable Traffic Server clients to access specific sites on the Internet without being authenticated by the LDAP server. Traffic Server uses a local database to improve the performance of LDAP authentication and, upon completion, logs successfully authenticated users.
- ✔ Secure connections in reverse proxy mode between a client and Traffic Server, and Traffic Server and the origin server, using the SSL termination option.
- ✔ Control access to the Traffic Manager UI using:
    - ✗ SSL (Secure Sockets Layer) protection for encrypted, authenticated access
    - ✗ An access control list (ACL) that defines which hosts are allowed to access the Traffic Manager
    - ✗ Administrator accounts that define which users can access the Traffic Manager and which activities they can perform (for example, view statistics only or view statistics and configure the Traffic Server)
- ✔ Set NNTP specific security options that:
    - ✗ Control user access to news articles cached by Traffic Server by defining access privileges for a particular group of clients
    - ✗ Enable external program-based NNTP authentication providing enterprise-wide control over news access, posting behavior, and other related privileges

Traffic Server security options are described in more detail in *Chapter 11, Security Options*.

Chapter 2

Getting Started

After you have installed Traffic Server on the nodes of your Traffic Server cluster, you are ready to begin using Traffic Server.

This chapter contains the following sections:

Starting Traffic Server

UNIX

In UNIX, you can start Traffic Server manually by issuing the `start_traffic_server` command. This command starts all the processes that work together to process Traffic Server requests and manage, control, and monitor the health of the Traffic Server system.

▼ To run the start_traffic_server command:

1 Log in to the node as the Traffic Server administrator and make Traffic Server's `bin` directory your working directory.

2 Enter the following command:

```
start_traffic_server
```

The Traffic Server starts.

Note Inktomi recommends that you always use the `start_traffic_server` command to start Traffic Server.

Windows

In Windows, you start Traffic Server by running the Inktomi Traffic Cop service. By default, the Inktomi Traffic Cop service is set to automatic so that it starts whenever Windows boots. If the Inktomi Traffic Cop service is set to manual, you must start it manually from the **Services** control panel.

▼ To start the Inktomi Traffic Cop service manually:

1 Open the Control Panel and double-click the **Services** icon.

2 Select the Inktomi Traffic Cop service, then click the **Start** button.
The Traffic Server starts.

Verifying that Traffic Server is up and running

After you have started Traffic Server for the first time, verify that it is up and running.

▼ To verify that Traffic Server is up and running:

1. Access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).
2. From the **Monitor** tab, click the **Protocols** button.
3. Make a note of the current **HTTP Client Total Document Bytes** statistic.
4. Set your browser to the Traffic Server proxy port.
5. Browse the internet.
6. Recheck the **HTTP Client Total Document Bytes** statistic.

   This value increases as the Traffic Server processes HTTP requests.

Accessing the Traffic Manager UI

The Traffic Manager UI is Traffic Server's browser-based user interface, consisting of a series of web pages. Traffic Manager provides a rich set of graphs and statistical displays for monitoring Traffic Server performance and network traffic, plus a set of options for configuring and fine-tuning your system.

You access the Traffic Manager through your web browser.

▼ To access the Traffic Manager:

1. Open your web browser.

   The Traffic Manager requires Java and JavaScript; be sure to enable Java and JavaScript in your browser.

2. Type one of the following locations in your browser:

   Standard

   ```
   http://nodename:adminport/
   ```

   SSL

   ```
   https://nodename:adminport/
   ```

   where `nodename` is the name of the Traffic Server node and `adminport` is the number assigned to the Traffic Manager port (the default value for `adminport` is 8081).

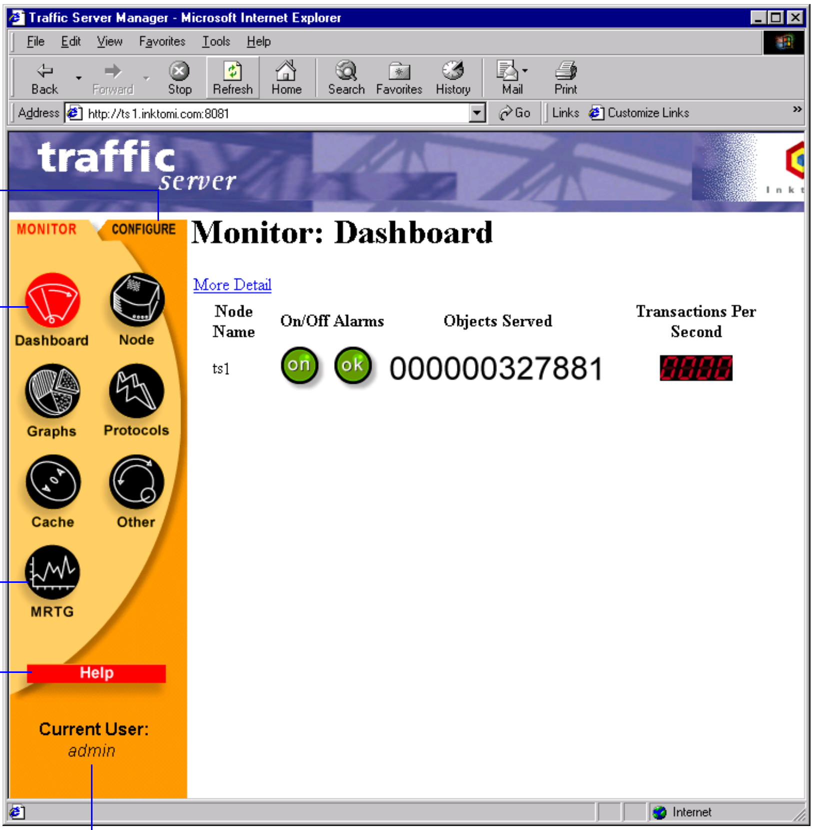
   Note Use the `https` address to reach the Traffic Manager only if you have restricted access to the Traffic Manager via SSL connections; otherwise, use the standard `http` address.

3. If necessary, log on to Traffic Server with the administrator ID and password, or your administrator account.

   The administrator ID and password are set during Traffic Server installation. You can change the ID and password, as well as create and modify administrator accounts. For more information, refer to *Controlling access to the Traffic Manager UI, on page 136*.

   The Traffic Manager UI opens in your web browser and displays the **Dashboard**, shown in *Figure 2, on page 33*.

.


Traffic Server Manager - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print
Address http://ts1.inktomi.com:8081
Go Links Customize Links
traffic server
MONITOR
CONFIGURE
Dashboard
Node
Graphs
Protocols
Cache
Other
MRTG
Help
Current User:
admin
Internet

Monitor: Dashboard

More Detail

Node Name	On/Off	Alarms	Objects Served	Transactions Per Second
ts1	on	ok	000000327881	0000

Click the Configure tab to display the Configure buttons and set configuration parameters

The Monitor tab contains seven buttons. Click a button to display a page of statistics

Click the Help button to display a description of the current page

This shows the current user logged in to Traffic Manager

Figure 2 The Monitor Dashboard

Using the Monitor and Configure tabs

The Traffic Manager UI has two tabs:

- ✔ The **Monitor** tab lets you view Traffic Server performance and network traffic statistics. For more information, refer to *Viewing statistics from Traffic Manager, on page 111*.
- ✔ The **Configure** tab lets you view and modify Traffic Server's configuration options. For more information, refer to *Chapter 10, Configuring Traffic Server*.

By default, the Traffic Manager UI starts by displaying the **Monitor** tab. To display the **Configure** tab, click the **Configure** tab to the right of the **Monitor** tab.

Using online help

Both the **Monitor** and **Configure** tabs provide a **Help** button. When you click the **Help** button, the Traffic Server online help opens in another browser window. The online help describes each page that opens when you click a button on the **Monitor** or **Configure** tab.

Starting Traffic Line

You can use Traffic Line to perform many of the tasks you can perform in the Traffic Manager UI.

Traffic Line has two command-line modes:

- ✔ Traffic Line batch mode

  You can use the batch mode to execute individual commands or to script multiple commands in a shell. Refer to *Appendix C, Traffic Line Commands* for a list of commands.

- ✔ Traffic Line interactive mode

  You can use interactive mode to retrieve statistics and to configure Traffic Server. Traffic Line interactive mode consists of several levels of commands. The Traffic Line interactive levels consist of the same commands available on the Traffic Manager **Monitor** and **Configure** tabs.

▼ To start a Traffic Line session:

1. In UNIX, log in to a Traffic Server node as the Traffic Server administrator and make Traffic Server's `bin` directory your working directory.

   In Windows, open a Command Prompt window, then change to the `bin` directory.

   You are now in a Traffic Line session and can enter Traffic Line commands.

   Traffic Line commands take the following form:

   ```
   traffic_line -flag argument
   ```

2. For a list of `traffic_line` commands, enter:

   ```
   traffic_line -h
   ```

3. To enter Traffic Line interactive mode, enter the following command:

   ```
   traffic_line -i
   ```

   For information about monitoring Traffic Server using Traffic Line interactive mode, refer to *Viewing Statistics from Traffic Line, on page 117*.

   For information about configuring Traffic Server using Traffic Line interactive mode, refer to *Configuring Traffic Server using Traffic Line, on page 128*.

Stopping Traffic Server

UNIX

In UNIX, you can stop Traffic Server by issuing the `stop_traffic_server` command.

Important Always use the `stop_traffic_server` command to stop the Traffic Server. Manually killing processes can lead to unpredictable results.

▼ To run the stop_traffic_server command:

1 Log in to the node as the Traffic Server administrator and make Traffic Server's `bin` directory your working directory.

2 Enter the following command:

```
stop_traffic_server
```

The Traffic Server stops.

Windows

In Windows, you stop Traffic Server by stopping the Inktomi Traffic Cop service.

▼ To stop the Inktomi Traffic Cop service manually:

1 Open the Control Panel and double-click the **Services** icon.

2 Select the Inktomi Traffic Cop service, then click the **Stop** button.
The Traffic Server stops.

Chapter 3

Web Proxy Caching

The idea behind web proxy caching is to store copies of frequently accessed documents close to users and serve this information to them on demand. Internet users get their information faster and Internet bandwidth is freed up for other tasks.

This chapter discusses the following topics.

Understanding web proxy caching

Internet users direct their requests to web servers all over the Internet. For a caching server to serve these requests, it must act as a *web proxy server.* A web proxy server fields user requests to arbitrary web servers and either serves the requests, or forwards them on to the *origin server* (the web server that contains the original copy of the requested information).

The Traffic Server proxy supports both *transparent proxy caching*, where the user's client software (typically a browser) is unaware that it is communicating with a proxy, and *explicit proxy caching*, where the user's client software must be configured to send requests directly to the traffic Server proxy.

A day in the life of a cache request

Here is an overview of the steps that take place as a Traffic Server proxy cache serves a user request.

Step 1 Traffic Server receives a user request for a document, image, news article, or other web object.

Step 2 With the object address in hand, Traffic Server looks up the requested object in its object database (cache).

Step 3 If the object is in the cache, Traffic Server checks to see if the object is fresh enough to serve. (See *Ensuring cached object freshness, on page 40* for details.) If the object is fresh, Traffic Server serves it to the user as a *cache hit* (*Figure 3*).

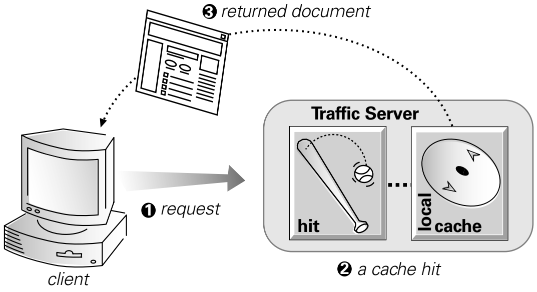


Figure 3 *A cache hit*

Step 4 If the data in the cache is stale, Traffic Server connects to the origin server and asks if the document is still fresh. If the document is still fresh, Traffic Server sends the cached copy to the user immediately.

Step 5 If the object is not in the cache (a *cache miss*) or the server indicates that the cached copy is no longer valid, Traffic Server gets the document from the origin server, simultaneously streaming it to the user and the cache (*Figure 4*). Subsequent requests for the object will be served faster.

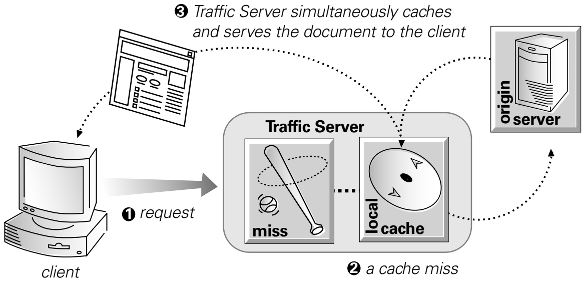

❸ Traffic Server simultaneously caches and serves the document to the client
origin server
Traffic Server
miss
local cache
❶ request
client
❷ a cache miss

Figure 4 *A cache miss*

Caching is more complex than the preceding overview suggests. In particular, the overview does not answer these questions:

- ✔ How does Traffic Server ensure freshness?
- ✔ How does Traffic Server serve correct HTTP alternates?
- ✔ How does Traffic Server treat requests for objects that cannot or should not be cached?

The following sections discuss these questions.

Ensuring cached object freshness

Traffic Server handles object freshness differently depending on protocol.

HTTP Web documents support optional author-specified expiration dates. Traffic Server adheres to these expiration dates; otherwise it picks an expiration date based on how frequently the document is changing and on administrator-chosen freshness guidelines. In addition, documents can be revalidated, checking with the server if a document is still fresh.

FTP FTP documents stay in the cache for a time period specified in the **Freshness** section of the **Cache** page in Traffic Manager's Configure Mode.

NNTP News articles are refreshed each time Traffic Server polls parent news servers for changes in group lists, article overview lists, and article updates. See *Maintaining the cache: updates and feeds, on page 48*.

Revalidating HTTP objects

If an HTTP object is stale, Traffic Server *revalidates* the object. A revalidation is a query to the origin server that asks if the object is unchanged. The result of a revalidation could be:

✔ The object is still fresh; Traffic Server resets its freshness limit and serves the object.

✔ A new copy of the object is available; Traffic Server caches the new object, replacing the stale copy, and serves the object to the user simultaneously.

✔ The object no longer exists on the origin server; Traffic Server does not serve the cached copy.

✔ The origin server does not respond to the revalidation query. The Traffic Server serves the stale object along with a `111 Revalidation Failed` warning.

HTTP object freshness tests

Here is how Traffic Server determines an HTTP document's freshness:

Expires header test Some documents come with `Expires` headers or `max-age` headers that explicitly define how long the document may be cached. A simple comparison of the current time with the expiration time tells Traffic Server whether or not the document is fresh.

Last-Modified / Date header test If there is no expiration information, a freshness limit can be estimated from the `Last-Modified` and `Date` headers. The `Last-Modified` header indicates how long ago a document was modified. If a document was last modified two years ago, it is unlikely to suddenly change, so Traffic Server can cache it safely for a while. But if the document just changed 5 minutes ago, it might be quite volatile, and Traffic Server should not cache it very long. Traffic Server stores an object for some percentage (F) of the time that elapsed since it last changed, 10% by default:

```
freshness limit = F * (Date - Last-Modified)
```

Where the `Date` header provides the date the object was sent to Traffic Server, and the `Last-Modified` header provides the date the object was last modified on the origin server.

For example, if a document was last modified 32 days ago and was sent to Traffic Server 2 days ago, it is considered fresh in cache for 3 days after it was sent, assuming a factor of 10%. It is considered fresh for one more day.

Because this method might select large freshness times for documents that have not changed for a long time, cache administrators may want to place an upper bound on the freshness limit. The freshness limit, then, is the minimum of this upper bound and the computed freshness limit. You configure this upper bound in the **Freshness** section of the **Configure: Cache** page of the Traffic Manager UI.

Default test For documents that do not have `Expires` headers or do not have both `Last-Modified` and `Date` headers, you can specify an absolute freshness limit in the **Freshness** section of the **Configure: Cache** page.

Revalidate rules in the cache.config file Revalidate rules apply specific freshness limits to specific HTTP or FTP objects. You can set freshness limits for objects originating from particular domains or IP addresses, objects with URLs that contain specified regular expressions, objects requested by particular clients, and so on. See *cache.config, on page 232*.

Deciding whether to serve HTTP objects

Even though a document may be fresh in the cache, clients or servers may have their own constraints that prevent them from retrieving the document from the cache. For example, a client might request that a document not come from a cache, or if it does, it cannot have been cached for more than 10 minutes.

Traffic Server bases the servability of a cached document on `Cache-Control` header fields. `Cache-Control` headers can appear in both client requests and server responses.

The following cache-control header fields affect whether objects are served:

- ✔ The `no-cache` field, sent by clients, tells Traffic Server to serve *no* objects directly from the cache; always revalidate. You can configure Traffic Server to ignore client no-cache fields.
- ✔ The `max-age` field, sent by servers, is compared to the document age; if the age is less than the `max-age`, the document is fresh and can be served.
- ✔ The `min-fresh` field, sent by clients, is an *acceptable freshness tolerance*. The client wants the object to be at least this fresh. If a cached document does not remain fresh at least this long in the future, it is revalidated.

- ✔ The `max-stale` field, sent by clients, permits Traffic Server to serve stale documents provided they are not too old. Some browsers may be willing to take slightly old documents in exchange for improved performance, especially during periods of poor Internet availability.

Traffic Server applies `Cache-Control` servability criteria *after* HTTP freshness criteria. For example, a document might be considered fresh, but if its age is greater than its `max-age`, it is not served.

Configuring HTTP freshness options

You can configure the following freshness guidelines for Traffic Server:

- ✔ How often to revalidate (when to consider objects stale). See *Configuring HTTP revalidation, on page 42*.
- ✔ Whether to cache documents without freshness information. See *Configuring HTTP cachability, on page 42*.
- ✔ The upper bound used to determine if the Last-Modified /Date freshness limit is too long.
- ✔ The absolute freshness lifetime used to estimate the freshness of documents without `Expires` or `Last-Modified` headers.

In the `cache.config` file, you can configure Traffic Server to revalidate objects from specific origin servers at specific times. Refer to *cache.config, on page 232*.

Configuring HTTP revalidation

The following HTTP revalidation options are available:

- ✔ Always revalidate (everything is considered stale).
- ✔ Never revalidate (everything is considered fresh).
- ✔ Revalidate all objects without `Expires` headers. Evaluate the freshness of objects with `Expires` headers by first checking the `Expires` header, and then checking `Cache-Control` headers.
- ✔ Evaluate freshness as follows:

1. Use the `Expires` header test, if applicable, otherwise go to step 2. If the object is stale, revalidate. If it is fresh, check the `Cache-Control` headers.
2. Use the `Last-Modified` / `Date` header test, if applicable, otherwise go to step 3. If the object is fresh according to the `Last-Modified` / `Date` test, check the `Cache-Control` headers for any freshness restrictions.
3. Use the absolute freshness limit specified in the **Freshness** section of the **Configure: Cache** page. Revalidate if the age is past the freshness limit.

Configuring HTTP cachability

The following HTTP cachability options are available:

- ✔ Cache only documents that have `Expires` headers
- ✔ Cache only documents that have `Expires` or `Last-Modified` headers
- ✔ Do not restrict caching

Caching HTTP alternates

Some origin servers answer requests to the same URL with a variety of objects. The content of these objects can vary widely, according to whether a server delivers content for different languages, targets different browsers with different presentation styles, or delivers variable content at different times of the day. Different versions of the same object are termed *alternates*.

Alternates are identified by header information. You can configure Traffic Server to cache all alternates according to a particular header. For example, if you tell Traffic Server to vary on the `User-Agent` header, Traffic Server caches all the different user-agent versions of documents it encounters. You can configure caching of alternates in the **Variable Content** section of the **Cache** page in Traffic Manager's Configure mode.

To cache or not to cache?

NNTP You can limit article caching to specific news groups. See *Blocking particular groups, on page 47*.

FTP You can specify never-cache rules for specific types of FTP documents in the `cache.config` file. See *cache.config, on page 232*.

HTTP Traffic Server responds to caching directives from clients and origin servers, as well as configurable options in the Traffic Manager UI and the `cache.config` file.

The following table lists the HTTP caching directives that Traffic Server follows.

Directive source	Caching directives

administration options	Traffic Server has the following administration options for caching:
	■ Configure Traffic Server not to cache objects with URLs containing the following:
	`?`
	`;`
	`/cgi`
	end in `.asp`
	■ Configure Traffic Server not to cache objects served in response to the Cookie: header.
	■ Use never-cache rules in the `cache.config` file. Refer to *cache.config, on page 232*.

client	Traffic Server does not cache objects with the following request headers. Note that some of these directives can be overridden by Traffic Server administration options. ■ Cache-Control: no-store header ■ Cookie: header ■ Authorization: header
origin server	Traffic Server does not cache objects with the following response headers. Note that some of these directives can be overridden by Traffic Server administration options. ■ Cache-Control: no-store ■ www-Authenticate: header ■ Set-Cookie: header ■ Cache-Control: no-cache header ■ Pragma: no-cache header ■ Expires: header with value of 0 (zero) or a past date

Scheduling updates to local cache content

To further increase the performance of Traffic Server, you can configure it to perform scheduled updates to the local cache content. This enables you to instruct Traffic Server to explicitly load specific objects into cache. You might find this especially beneficial when using Traffic Server as a reverse proxy for server acceleration, enabling you to preload content that you anticipate will be in demand. See *Understanding reverse proxy caching, on page 66*.

You do this by inputting a list of URLs in the **Content Management** page in the Traffic Manager UI. This modifies the `update.config` file, specifying objects that you want to schedule for update along with the time and interval of when this update should take place. The UI also enables you to specify a recursion depth for the URL.

Traffic Server uses this information to determine the URLs for which it is responsible and, for each URL, derives all recursive URLs if applicable. It then generates a unique, sorted URL list. Using this list, Traffic Server initiates an HTTP `GET` for each un-accessed URL, ensuring that it remains within the user-defined limits for HTTP concurrency at any given time. The system logs the completion of all HTTP `GET` operations, enabling you to monitor the performance of this feature.

News article caching

Traffic Server can function as a news server or a caching news server. This section provides background information about Traffic Server news server and the Network News Transfer Protocol (NNTP) caching features.

News, also known as USENET and *discussions*, is a system of online discussion groups. NNTP is the protocol used to retrieve and distribute these discussion groups. News groups exist to discuss just about any subject; for example, `rec.humor`, `talk.religion`, `news.answers`, `rec.food.recipes`, and `comp.std.unix`. The articles posted to these groups are propagated around the world. Traffic Server supports NNTP as specified in RFC 977 and many common extensions and proposed extensions.

To read news articles, users need a news reader, such as Netscape Communicator or Microsoft Internet Explorer, and access to a news server. Traffic Server is a caching news server. It can be configured to sit transparently between users and a *parent* or *backing* news server, increasing responsiveness for the user and decreasing network bandwidth use and the load on the parent news server.

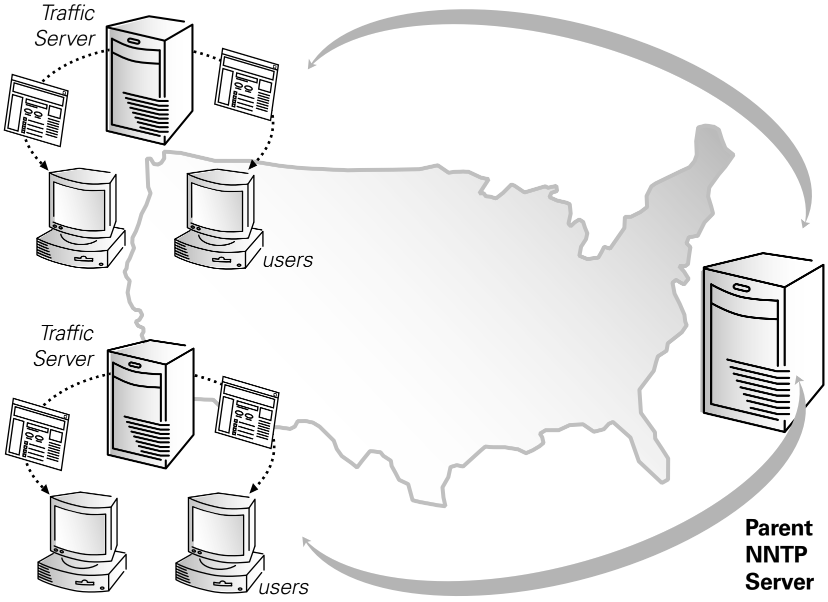


Figure 5 Traffic Servers caching news articles for a distant NNTP server

Traffic Server provides many configurable options for supporting parent NNTP servers. The following sections describe Traffic Server's NNTP features.

Traffic Server as a news server

When clients want to read news, they access a news server. The news server offers a list of groups to which clients can subscribe. For each subscribed group, the clients read an overview list of the articles in the group, and then select an article to read. When Traffic Server acts as a news server, it:

- ✔ Maintains lists of supported news groups
- ✔ Accepts news feeds for each supported news group
- ✔ Serves requested articles to users
- ✔ Accepts and numbers user postings to its supported news groups

Traffic Server as a caching proxy news server

When Traffic Server acts as a caching proxy news server for a particular news server, it:

- ✔ Maintains lists of the news groups on its parent NNTP servers. You can configure the frequency that Traffic Server updates its copies of group lists.
- ✔ Caches and serves article overview lists on demand. You can also tell Traffic Server to pull article overview lists from the parent news server periodically.
- ✔ Caches and serves articles on demand. Traffic Server can also accept news feeds, like any news server.
- ✔ Caches and serves miscellaneous *LIST* files, such as subscription files.
- ✔ Sends user postings to the parent news server.

When clients issue news requests, Traffic Server intercepts these requests and serves them from its cache, reducing traffic to parent news servers. If a particular overview or article is not in the cache, Traffic Server forwards requests to the parent server.

Supporting several parent news servers

Traffic Server can cache articles for several news servers. You specify all of the parent news servers for Traffic Server in the `nntp_servers.config` file. See *nntp_servers.config, on page 252*. For each parent news server, you can have Traffic Server cache some or all of that server's news groups.

Some of the possible parent configurations that Traffic Server supports are as follows:

Several news servers supplying the same groups

Several news servers can be configured to redundantly serve the same groups, providing enhanced reliability. Traffic Server provides the following features for managing these configurations:

- ✔ Priorities

  If Traffic Server has to contact a parent news server for information about a group supplied by several news servers, Traffic Server contacts the news server with the highest priority.

- ✔ Round-robin

  If several parent news servers supplying the same group have the same priority, Traffic Server selects a parent news server in round robin fashion.

- ✔ Failover

  If a request to a parent server fails, Traffic Server tries the next server in the round robin (of the same priority), and then servers of lower priority.

- ✔ Background retries

  Failed servers are retried in the background and are used (restored to their specified priority) when they become available.

Several servers supplying different groups

Several news servers can be configured with news servers supplying different (disjoint) groups. Administrators can use this feature to spread the load based on group.

NNTP cache hierarchies

Using a Traffic Server as parent to a group of Traffic Servers can reduce load on a parent news server and take advantage of the large number of concurrent connections Traffic Server supports.

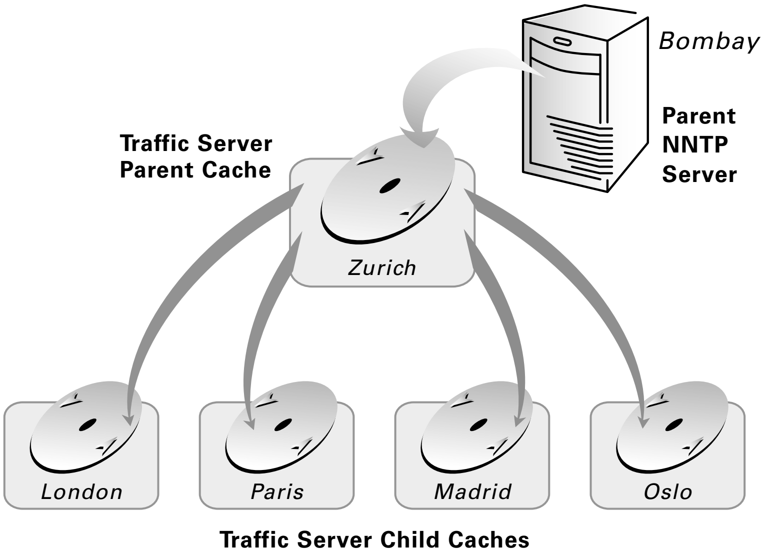


Figure 6 Hierarchy of news caching servers

In *Figure 6* above, the *parent news server* for each of the child Traffic Servers is the parent Traffic Server. The parent Traffic Server is a child cache to the distant parent news server.

Nonstandard ports and network interfaces

You can configure the interface from which to connect to a parent news server port. You can also configure the port on the parent server to which Traffic Server connects.

Blocking particular groups

You can block particular groups on specified news servers. Clients do not see blocked groups in news server group lists. You list all blocked groups in the `nntp_servers.config` file; see *page 252*.

Clustering

Articles, overview lists, group lists, and LIST files are all maintained in Inktomi's high performance object store. This information is updated at configurable intervals so that users and child caches see a consistent view of news.

Large clusters of Traffic Servers can be configured to act as a single large virtual cache with all the storage and serving power of the aggregate. See *Chapter 6, Traffic Server Clusters*. Article numbers and group information are maintained consistently across the cluster.

Transparency

NNTP traffic bound for a well known NNTP server can be intercepted transparently by Traffic Server. By transparently intercepting, caching, and serving the NNTP data from a centralized parent news server, Traffic Server simplifies migration and administration while increasing responsiveness and decreasing network utilization.

To run Traffic Server in NNTP transparent mode, you must enable NNTP transparency during Traffic Server installation. See the *Traffic Server Installation Guide* for more information. If you do *not* have transparency enabled on your Traffic Server but choose to enable NNTP, it will run in non-transparent mode and news readers must explicitly configure Traffic Server as their news server.

Posting

Traffic Server sends user article postings to the parent news server. You can specify the parent news server that receives postings for a particular group or set of groups in the `nntp_servers.config` file; see *page 252*. When Traffic Server acts as the news server (accepting article feeds), Traffic Server accepts postings.

With background posting, the Traffic Server queues posted articles until the posting news server can accept the posted article.

Maintaining the cache: updates and feeds

Traffic Server can maintain the freshness of its cache by:

- ✔ Updating its cache on demand
- ✔ Actively retrieving (*pulling*) updates at configurable intervals
- ✔ Accepting news feeds

The `nntp_servers.config` file controls the Traffic Server's caching behavior for specific news groups; see *page 252* for more information. You configure update frequencies in the **Configure: Protocols** page of the Traffic Manager UI. Here are the available options:

- ✔ Pull the overview information for specified groups

  For all groups designated as *pullover*, the server will retrieve the overview database information (using the `OVER`/`XOVER` commands) automatically and periodically.

  Pulling overview information can be useful for high volume groups which are frequently read but from which only a subset of the articles are accessed.

- ✔ Pull the articles for specified groups

  For all groups designated as *pull*, Traffic Server retrieves the articles automatically and periodically.

  Pulling groups is useful when the administrator does not wish to or cannot set up a full or partial feed.

- ✔ Dynamically subscribe to specified groups

  Traffic Server can monitor the usage pattern for groups, and those for which the overview database is very frequently accessed can be treated as pullover groups. Likewise those for which the articles are very frequently accessed can be treated as pull groups.

✔ Take a partial feed (push) for specified groups

For all groups designated as *push*, Traffic Server verifies that it has any requested articles and retrieves them from the parent server if they are not available locally.

Partial feeds are useful for groups where *some* articles are always accessed, or for shifting article transport to a time of day when bandwidth is cheaper or underutilized.

✔ Take a full feed for some or all groups

For all groups designated as *feed*, Traffic Server does not connect to the parent news server, and instead acts like a conventional news server. In particular, if a cache miss occurs, Traffic Server does not forward the request to a parent news server.

Full feeds can be used for very high volume groups in which most or all the articles are accessed or for shifting article transport to a time when bandwidth is cheaper or more plentiful.

Caution Taking a full feed is not recommended as the server will have no way to retrieve an article if it is lost for any reason (such as lack of space or hardware failure).

Configuring access control

You can configure different types of user authentication based on source domain, hostname, or IP range. These values are set in the `nntp_access.config` file. Here are the available options:

Option	**Description**
Allow or deny	You can simply allow or deny particular domains, hosts, or IP ranges.
Basic	This option is simple authentication based on user name and optional password.
Generic	Generic authentication allows a specified program on the authentication server (which can be the Traffic Server host machine or a specified remote authentication server) to communicate with an arbitrary program on the client to do the authentication.
Custom	An arbitrary program on the authentication server (which can be the Traffic Server host machine or a specified authentication server) can be used to do the authentication based on the client hostname, client IP, and optionally the client user name and password. You can use custom authentication to interface Traffic Server to any standard or homegrown access control mechanism.
Authentication server	An authentication server, possibly located on a different host machine, can be used to do the authentication (generic and custom) This enables authentication to be centralized.
Version 2 NNTP authentication	Traffic Server supports version 2 NNTP authentication. Do not use this form of authentication unless you are certain that all of your clients use version 2 authentication.

Using enhanced NNTP authentication

Traffic Server enables you to leverage your existing directory infrastructure to perform enhanced NNTP authentication. You might consider using this feature in cases where simple user authentication, based on source domain, hostname, or IP range, is insufficient for the needs of your organization. For example, you might need to base your NNTP access control policies on factors such as time of day, system load, or other dynamic characteristics.

Traffic Server supports enhanced NNTP authentication through the use of external programs, known as plugins, which interface directly with your directory or database. You can write or script plugins using any language that permits you to connect to a directory or database, including PERL and C, among others. You can then have plugins reside on the same host as the Traffic Server, or you can position them anywhere on the network, connected by a secure tunnel if required.

Each time a user needs to be authenticated, Traffic Server connects to the authentication server, which is part of Traffic Server, and runs the plugin.

Obeying NNTP control messages

The Traffic Server default setup for nonfeed news groups is to periodically check the parent server for new groups, cancelled articles, and new articles. If you have enabled these periodic checks in the **Configure: Protocols** page, you do not need to enable obeying control messages.

Traffic Server can be configured to obey NNTP control messages. In particular, you can enable Traffic Server to obey `cancel`, `addgroup`, and `rmgroup` messages in the **Configure: Protocols** page of the Traffic Manager UI. For example, if you select “Obey cancel control messages,” Traffic Server pulls `cancel` messages automatically in order to obey them.

Client bandwidth throttling

You can limit the amount of bandwidth allotted to clients for downloading articles. Clients that attempt to exceed the bandwidth limit will have each operation slowed in order to keep their bandwidth consumption to the limit. You set the bandwidth limit (the client speed throttle) in the **NNTP** section of the **Protocols page** in Traffic Manager Configure mode.

Chapter 4

Transparent Proxy Caching

The transparency option enables Traffic Server to respond to Internet requests without requiring users to reconfigure their browser settings.

This chapter discusses the following topics.

Serving requests transparently

In non-transparent operations, client browsers must be configured to send web requests to the Traffic Server proxy cache. Many sites have no direct control over user browser settings, making it necessary for site administrators to tell users to configure their browsers to direct requests to a Traffic Server.

The transparency option enables Traffic Server to respond to Internet requests without requiring users to reconfigure their browser settings. It does this by redirecting the traffic flow into the Traffic Server cache after it has been intercepted by an L4 switch or router.

Here's how Traffic Server transparent interception works:

Step 1 Traffic Server intercepts client requests to origin servers. There are several ways to deploy Traffic Server so that interception can take place. See *Interception strategies, on page 53* for details.

Step 2 Inktomi's Adaptive Redirection Module (ARM) redirects requests destined for origin servers to the Traffic Server application. See *ARM redirection, on page 59* for details.

Step 3 A very small number of clients and servers do not work correctly through proxies. Traffic Server identifies these problem clients and servers dynamically, and the ARM adaptively disables interception for these clients and servers, passing their traffic unimpeded to the origin server. Additionally, clients and servers can be manually exempted from caching by configuring ARM. See *Interception bypass, on page 60* for more information.

Step 4 Traffic Server receives and begins processing the intercepted client requests as usual. If a request is a cache hit, Traffic Server serves the requested document or news article. If a request is a miss, Traffic Server retrieves the document from the origin server and serves it to the client.

Step 5 On the way back to the client, the ARM changes the source IP address to the origin server IP address and the source port to the origin server port.

Interception strategies

The transparency routing solutions supported by Traffic Server are:

- ✔ A Layer 4-aware switch.

  See *Using a layer 4-aware switch to filter transparency requests* below.
- ✔ A Cisco IOS-based router using the Web Cache Control Protocol (WCCP).

  See *Using a WCCP-enabled router for transparency, on page 54*.
- ✔ Policy-based routing.

  See *Using policy-based routing to filter transparency requests, on page 57*.
- ✔ Software routing.

  See *Software-based transparency solutions, on page 58*.

How client requests reach Traffic Server depends on network topology. In a complex network, you must decide which clients are to be served transparently and make sure that Traffic Server is positioned to intercept their requests. Traffic Server, or routers or switches feeding Traffic Server, are often deployed at a major artery or aggregation pipe to the Internet.

You configure and enable transparency options during Traffic Server installation. See the *Traffic Server Installation Guide* for more information about enabling transparency.

Using a layer 4-aware switch to filter transparency requests

- ✔ Layer 4-aware switches have the ability to rapidly redirect supported protocols to Traffic Server, while passing all other Internet traffic through directly to its destination. *Figure 7* below illustrates this scenario for HTTP.

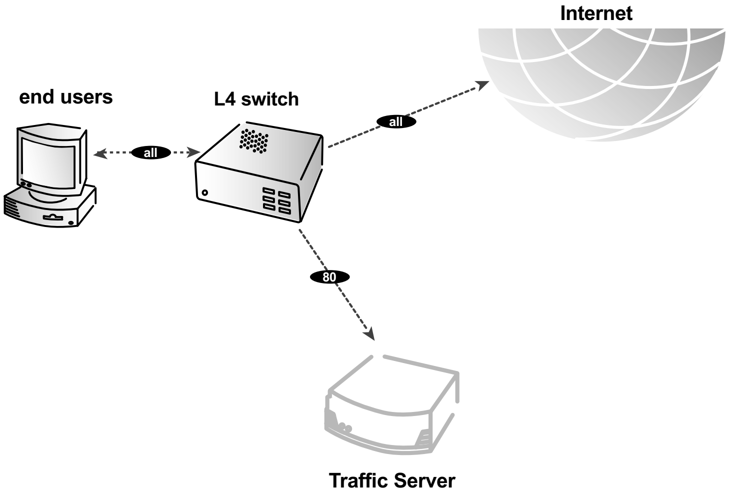


Figure 7 Using a layer 4-aware switch to filter HTTP requests

Layer 4-aware switches offer the following features, depending on the particular switch:

- ✔ A layer 4-aware switch that can sense downed hosts on the network and redirect traffic adds reliability.
- ✔ If a single layer 4-aware switch feeds several Traffic Servers, the switch handles load balancing among the Traffic Server nodes. Different switches might use different load balancing methods, such as round-robin or hashing. If a node becomes unavailable, the switch automatically redistributes the load. When the node returns to service, some switches automatically return the node to its previous workload, so that the node cache need not be repopulated; this feature

is called *cache affinity*. Inktomi recommends that you do *not* enable Traffic Server virtual IP failover in this situation, because layer 4-aware switch failover is already in operation.

Using a WCCP-enabled router for transparency

Traffic Server supports WCCP 1.0 and WCCP 2.0.

A WCCP 1.0-enabled router can send all port 80 (HTTP) traffic to Traffic Server, as shown in *Figure 8* below. The Traffic Server ARM readdresses port 80 to Traffic Server's proxy port (by default, port 8080). Traffic Server processes the request as usual, retrieving the requested document from the cache if it is a hit and sending the response back to the client. Along the way, the ARM readdresses the proxy port in the response header to port 80 (undoing the readdressing it did on the way to Traffic Server). The user then sees the response exactly as if it were sent directly from the origin server.

A WCCP 2.0-enabled router works in the same way as a WCCP 1.0-enabled router. In addition to port 80 (HTTP) traffic, WCCP 2.0 supports additional protocols including NNTP (port 119 traffic).

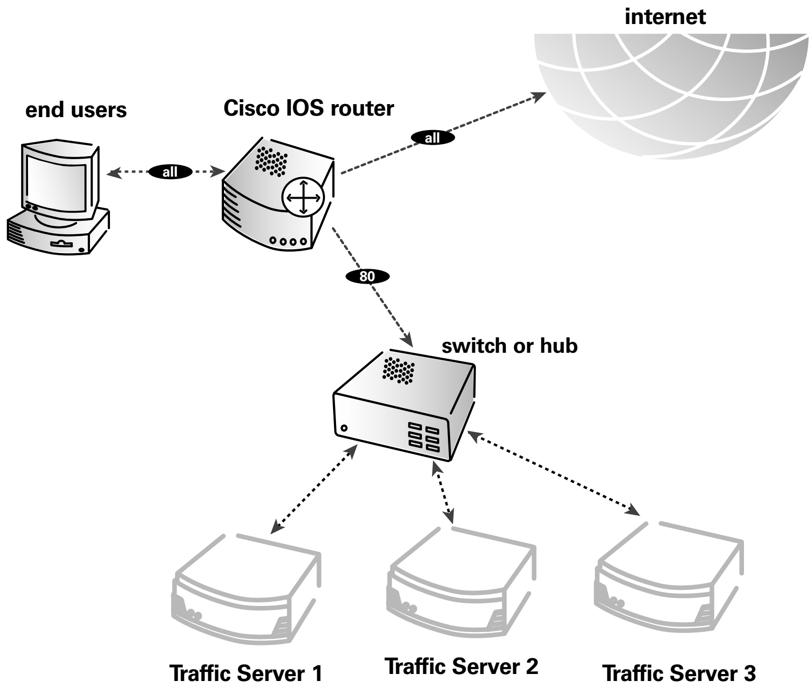


Figure 8 Using a Cisco IOS router to send port 80 traffic to several Traffic Servers

WCCP provides the following routing benefits:

- ✔ The WCCP-enabled router and Traffic Server exchange heartbeat messages, letting each other know they are running. The WCCP router automatically reroutes port 80 traffic (and port 119 traffic in WCCP 2.0) if the Traffic Server goes down.
- ✔ If several Traffic Servers receive traffic from a WCCP router, WCCP balances the load among the Traffic Servers. The group of Traffic Servers is called a *WCCP cache farm*. See *About WCCP load balancing, on page 56*.
- ✔ Traffic Server handles node failure in WCCP cache farms. If one node becomes unavailable, its load is redistributed among the remaining nodes.

✔ In WCCP 2.0, you can use multiple routers. Traffic flowing through multiple routers can share the same pool of caches.

Enabling WCCP on Traffic Server

Typically, you enable WCCP when you install Traffic Server (refer to the *Traffic Server Installation Guide* for installation instructions). However, you can enable WCCP on Traffic Server at any time after installation by editing the `records.config` file.

Important Before you enable WCCP, make sure that your configuration meets the following requirements:

✔ Traffic Server 2.1 or newer is running on Solaris systems for WCCP 1.0. Traffic Server 3.0.1 or newer is running on Solaris systems for WCCP 2.0.

✔ The WCCP 1.0 router is running Cisco IOS Release 11.1(18)CA or 11.2(13)P or newer. The WCCP 2.0 router is running Cisco IOS Release 12.0(3)T or newer. Check Cisco System's home page for a list of the platforms that support WCCP.

✔ If you are using several Traffic Server nodes, determine whether you want the Traffic Server nodes to have management-only clustering or full clustering (refer to *About WCCP load balancing, on page 56*).

✔ Each Traffic Server must have the transparency option installed. Refer to the *Traffic Server Installation Guide*.

✔ WCCP must be enabled on the router that is sending traffic to Traffic Server. Instructions for enabling WCCP on Cisco routers is provided on Cisco System's home page.

The following procedures describe how to enable WCCP on Traffic Server after installation. Different procedures are provided for WCCP 1.0 and WCCP 2.0. Follow the procedure appropriate for your environment.

▼ To enable WCCP 1.0 after installation:

1. In a text editor, open the `records.config` file located in the `config` directory.
2. Set the following variable to 1:

```
proxy.config.wccp.enabled INT 1
```

3. Edit the following variable to specify the IP address of the WCCP router that is sending traffic:

```
proxy.config.wccp.router_ip STRING router_IP_address
```

4. Save and close the `records.config` file.
5. Make Traffic Server's `bin` directory your working directory and run the command `traffic_line -x` to apply the configuration changes.

▼ To enable WCCP 2.0 after installation:

1. In a text editor, open the `records.config` file located in the `config` directory.
2. Set the following variable to 1:

```
proxy.config.wccp.enabled INT 1
```

3. Set the following variable to 2:

```
proxy.config.wccp.version INT 2
```

Optional 4. To enable security so that Traffic Server and your routers can authenticate each other, set the following variable to 1:

```
proxy.config.wccp2.security_enabled INT 1
```

5 For unicast mode, go to *step 7*.

For multicast mode, set the following variable to 1:

```
proxy.config.wccp2.multicast_enabled INT 1
```

6 Edit the following variable to specify the IP multicast address:

```
proxy.config.wccp2.multicast_address STRING address
```

7 If multicast mode is *not* enabled, you must edit the following variable to specify the number of routers that direct traffic to Traffic Server.

```
proxy.config.wccp2.number_of_routers INT number
```

8 If multicast mode is *not* enabled, you must edit the following variable to specify the IP addresses of each router that directs traffic to Traffic Server:

```
proxy.config.wccp2.router0_ip STRING IPaddress
proxy.config.wccp2.router1_ip STRING IPaddress
...
```

9 Save and close the `records.config` file.

10 Make Traffic Server's `bin` directory your working directory and run the command `traffic_line -x` to apply the configuration changes.

Tip To check that the router is sending traffic to Traffic Server, look at the statistics in the **Monitor** pages of Traffic Manager. For example, check that the **Objects Served** value in the **Dashboard** page increases.

ARM bypass and WCCP

If Traffic Server has an ARM bypass rule (discussed in *Interception bypass, on page 60*), Traffic Server forwards particular client requests directly to the origin server, bypassing the cache. Bypassed requests are unchanged by the ARM; they retain their client source IP addresses. In WCCP 1.0, ARM bypass rules cannot work if the WCCP router is also Traffic Server's default gateway router, as shown in *Figure 8, on page 54*. The WCCP router sends port 80 traffic to the Traffic Servers *and* it serves as the Traffic Servers' default gateway or *next hop* to the internet. Bypassed requests go to the WCCP router, which sends them back to Traffic Server.

In WCCP 2.0, you can exclude certain router interfaces from redirection. Traffic Server bypass rules can work if you exclude the router interface on which Traffic Server is connected from using WCCP. You can do this by setting the router configuration command `ip wccp redirect exclude in` (refer to Cisco's WCCP documentation for information about router configuration).

About WCCP load balancing

If a WCCP router serves several nodes, as in *Figure 8, on page 54* the router balances load among the Traffic Servers. The router sends each node requests aimed at a particular range of IP addresses, so that each node is responsible for caching content residing at particular IP addresses.

You can monitor the percentage of traffic that goes to each node. If a node becomes unavailable, its traffic is redistributed.

Traffic Server also supports cache affinity. If a node becomes unavailable and then recovers, Traffic Server returns the node to its former load distribution. This means that the node's cache need not be repopulated.

The WCCP cache farm acts as a simple form of distributed cache, which is sufficient for many applications. A WCCP-enabled network device distributes traffic to individual Traffic Servers based on the IP address of the origin server. Each node caches objects requested from a particular set of origin servers, which belong to that node's assigned range of destination IP addresses.

Traffic Server's full clustering option is not required for WCCP and you can run Traffic Server nodes in management-only clustering mode. During Traffic Server installation, if you select clustering and enable WCCP, management-only clustering is enabled by default. Management-only clustering conserves CPU resources, and slightly improves performance over full clustering. See *Chapter 6, Traffic Server Clusters* for details.

Busy origin servers are often mapped to several IP addresses (using a DNS round-robin mechanism). Using WCCP-based load balancing alone, each of these different IP addresses could be allotted to different Traffic Server nodes. This can result in a slightly lower hit rate and wasted cache space, since the same content is being replicated across nodes. Traffic Server's full clustering mode ensures that all requests to a specific page on that origin server (no matter which IP address is used) are cached on the same node.

With full clustering, objects are distributed among nodes according to their URLs; WCCP distributes objects according to destination IP address. If a particular IP address is receiving many requests, WCCP load balancing may lead to a hot spot, where all of that site's traffic is cached on one node, instead of being distributed among the nodes. Traffic Server's full-clustering mode distributes different pages from the busy site to different Traffic Server nodes.

In general, if load-handling capacity and latency are most important, Inktomi recommends management-only clustering in WCCP environments. If hit rate, bandwidth savings, and better load balancing are most important, then full clustering may provide an improvement in WCCP environments.

If you are running clustered Traffic Servers, Inktomi recommends that you do *not* enable virtual IP failover in WCCP environments. Traffic Server's WCCP failover mechanism handles node failures and restarts. See *Virtual IP failover, on page 84* for details about virtual IP failover.

Using policy-based routing to filter transparency requests

Instead of the WCCP protocol, you can use the policy routing capabilities of a router to send traffic to Traffic Server. WCCP or an L4 switch are generally preferable to this configuration because policy-based routing has a performance impact on the router, and policy-based routing does not support load balancing or heartbeat messaging. *Figure 9, on page 58* illustrates this scenario for HTTP.

✔ All client Internet traffic is sent to a router that feeds Traffic Server.

✔ The router sends port 80 (HTTP) traffic to Traffic Server and sends the remaining traffic to the next hop router.

✔ The ARM translates intercepted requests into Traffic Server requests.

✔ Translated requests are sent to Traffic Server.

✔ Web documents to be served transparently are readdressed by the ARM on the return path to the client, so that the documents appear to have come straight from the origin server.

A Traffic Server cluster with virtual IP failover adds reliability; if one node fails, another node can take up its transparency requests. See *Virtual IP failover, on page 84*.

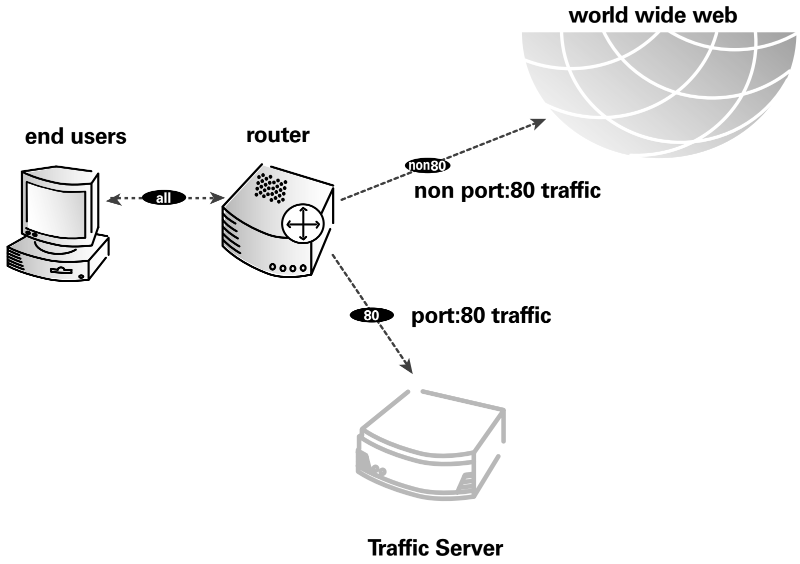


Figure 9 Using a router to filter HTTP requests

Software-based transparency solutions

You can deploy Traffic Server transparently without adding routers or switches by using the Traffic Server node as a software router and directing all traffic through Traffic Server. This solution can be useful in low-traffic situations, where the performance cost of using the Traffic Server machine as a router is not high. Traffic Server supports two software routing packages, `routed` and `gated`, that support the use of Traffic Server as a full intercepting router.

- ✔ All Internet traffic goes through Traffic Server from machines behind it in the network.
- ✔ The routing software, in this case `gated` or `routed`, routes all non-transparency requests out to the Internet; it routes port 80 HTTP requests to Traffic Server.
- ✔ The ARM translates intercepted requests into Traffic Server requests.
- ✔ Translated requests are sent to Traffic Server.
- ✔ Web documents to be served transparently are readdressed by the ARM on the return path to the client, so that the documents appear to have come straight from the origin server.

The `routed` software package included with Sun Solaris systems uses the Reverse Internet Protocol (RIP) and is adequate for simple routing needs. To install `routed`, see your Sun Solaris documentation.

The `gated` daemon is an extensible commercial software package from the Merit GateD Consortium. The `gated` daemon can replace `routed`. Moreover, `gated` supports several IP network routing protocols (RIP, BGP, and OSPF) and allows policy-based `routed` filtering. For instructions on installing `gated`, see the *Traffic Server Installation Guide*.

If you are considering using a software-based transparency solution, note that:

- ✔ While Traffic Server host machines can function as routers, they are not expressly designed to be routers.
- ✔ For reliability, you can use a Traffic Server cluster with the virtual IP failover option. If one node fails, another cluster node takes over. The Traffic Server cluster failover mechanism is similar to the hot standby router protocol (HSRP).

ARM redirection

The intercepted client requests that reach Traffic Server are addressed to an origin server. Traffic Server's ARM readdresses requests to Traffic Server so that they can be served.

Transparency cannot be enabled without the ARM. If you want to run Traffic Server transparently, *you must install the ARM* during Traffic Server installation. See the *Traffic Server Installation Guide*.

The ARM can make two changes to an incoming packet's address: its destination IP address and its destination port. Typically, HTTP packet destination IPs and ports are readdressed with the IP address of Traffic Server, and Traffic Server's HTTP proxy port (usually port 8080). NNTP packet destination IPs are readdressed with the IP address of Traffic Server. If Traffic Server uses a port other than 119 for NNTP, the destination NNTP port is readdressed as well. You can configure packet readdressing in the `ipnat.conf` file. This file contains redirection rules that specify how incoming packets should be readdressed; see the *Traffic Server Installation Guide* for more information.

Interception bypass

A very small number of clients and servers do not interoperate correctly with web proxies. Some of the causes of interoperability problems include:

- ✔ Client software bugs (homegrown, non-commercial browsers)
- ✔ Server software bugs
- ✔ Applications which send non-HTTP traffic over HTTP ports as a way of defeating security restrictions
- ✔ Server IP authentication (the origin server limits access to a few client IP addresses, but the Traffic Server IP address is different, so it cannot get access). This is not in frequent use because many ISPs dynamically allocate client IP dial-up addresses, and more secure cryptographic protocols are now more often used.

Web proxies are very common in corporate and Internet use, so the frequency of interoperability problems is extremely rare. However, Traffic Server contains an adaptive learning module that recognizes interoperability problems caused by transparent proxying and automatically bypasses the traffic around Traffic Server without operator intervention.

Traffic Server follows two types of bypass rules:

- ✔ *Dynamic* (also called *adaptive*) bypass rules are generated dynamically if you configure Traffic Server to bypass the cache when it detects non-HTTP traffic on port 80, or when it encounters certain HTTP errors. See *Dynamic bypass rules* below.
- ✔ *Static* bypass rules must be manually configured in the a bypass configuration file (`bypass.config`). See *Static bypass rules, on page 63*.

Note Do not confuse bypass rules with client access control lists. Bypass rules are generated in response to interoperability problems. Client access control is simply restriction of the client IP addresses that can access the Traffic Server cache as described in *Controlling client access to the Traffic Server proxy cache, on page 133*.

Dynamic bypass rules

When configured to do so, Traffic Server watches for certain protocol interoperability errors, and as it detects errors, it configures the ARM to bypass the proxy for those clients and/or servers causing the errors.

In this way, the very small number of clients or servers that do not operate correctly through proxies are auto-detected and routed around the Traffic Server, so they can continue to function normally (but without the improvement of caching).

You can configure Traffic Server to dynamically bypass the cache for any of the following triggering conditions:

Error code	Description
N/A	non-HTTP traffic on port 80
400	Bad Request
401	Unauthorized
403	Forbidden (authentication failed)
405	Method not allowed
406	Not Acceptable (access)
500	Internal server error

For example, when Traffic Server is configured to bypass on authentication failure (`403 Forbidden`), if any request to a host returns a 403 error, the ARM generates a destination bypass rule for the host's IP address. All requests to that host are bypassed until the next Traffic Server restart.

In another example, if the ARM detects that a client is sending a non-HTTP request on port 80 to a particular origin server, the ARM generates a source/destination rule. All requests from that particular client to the origin server are bypassed; requests from other clients are not bypassed.

Bypass rules that are generated dynamically are purged after a Traffic Server restart. If you want to preserve dynamically generated rules, you can save a snapshot of Traffic Server's current set of bypass rules. See *Viewing the current set of bypass rules, on page 64*.

Setting dynamic bypass rules

By default, Traffic Server is not configured to bypass the cache when it encounters HTTP errors or non-HTTP traffic on port 80. You must enable dynamic bypass rules in the `records.config` file.

▼ To set dynamic bypass rules:

1 In a text editor, open the `records.config` file located in the `config` directory.

2 Edit the following variables in the `ARM (Transparency Configuration)` section of the file:

Variable	Description
proxy.config.arm.bypass_dynamic_enabled	Set this variable to 1 to enable dynamic bypass.
proxy.config.arm.bypass_use_and_rules_bad_client_request	Set this variable to 1 to enable dynamic source/destination bypass in the event of non-HTTP traffic on port 80.
proxy.config.arm.bypass_use_and_rules_400	Set this variable to 1 to enable dynamic source/destination bypass when an origin server returns a 400 error.
proxy.config.arm.bypass_use_and_rules_401	Set this variable to 1 to enable dynamic source/destination bypass when an origin server returns a 401 error.
proxy.config.arm.bypass_use_and_rules_403	Set this variable to 1 to enable dynamic source/destination bypass when an origin server returns a 403 error.
proxy.config.arm.bypass_use_and_rules_405	Set this variable to 1 to enable dynamic source/destination bypass when an origin server returns a 405 error.
proxy.config.arm.bypass_use_and_rules_406	Set this variable to 1 to enable dynamic source/destination bypass when an origin server returns a 406 error.
proxy.config.arm.bypass_use_and_rules_408	Set this variable to 1 to enable dynamic source/destination bypass when an origin server returns a 408 error.
proxy.config.arm.bypass_use_and_rules_500	Set this variable to 1 to enable dynamic source/destination bypass when an origin server returns a 500 error.
proxy.config.arm.bypass_on_bad_client_request	Set this variable to 1 to enable dynamic destination bypass in the event of non-HTTP traffic on port 80.

Variable	Description
proxy.config.arm.bypass_on_400	Set this variable to 1 to enable dynamic destination bypass when an origin server returns a 400 error.
proxy.config.arm.bypass_on_401	Set this variable to 1 to enable dynamic destination bypass when an origin server returns a 401 error.
proxy.config.arm.bypass_on_403	Set this variable to 1 to enable dynamic destination bypass when an origin server returns a 403 error.
proxy.config.arm.bypass_on_405	Set this variable to 1 to enable dynamic destination bypass when an origin server returns a 405 error.
proxy.config.arm.bypass_on_406	Set this variable to 1 to enable dynamic destination bypass when an origin server returns a 406 error.
proxy.config.arm.bypass_on_408	Set this variable to 1 to enable dynamic destination bypass when an origin server returns a 408 error.
proxy.config.arm.bypass_on_500	Set this variable to 1 to enable dynamic destination bypass when an origin server returns a 500 error.

Important For a dynamic source/destination bypass rule to work, you must also enable the equivalent destination bypass rule. For example, when you set the variable `proxy.config.arm.bypass_use_and_rules_403` to 1, you must also set the variable `proxy.config.arm.bypass_on_403` to 1.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

Viewing dynamic bypass statistics

Traffic Server tallies bypassed requests for each type of dynamic bypass trigger. For example, Traffic Server counts all requests that are bypassed in response to a 401 error.

You can view the dynamic bypass statistics from Traffic Line.

▼ To view dynamic bypass statistics:

1 In UNIX, log on to a Traffic Server node as the Traffic Server administrator, then make Traffic Server's `bin` directory your working directory.

In Windows, open a Command Prompt window, then `cd` to the `bin` directory (located in the Traffic Server installation directory).

2 Enter the following command, and then press Return:

```
traffic_line -i
```

3 Enter the following command, and then press Return:

```
get variable
```

where variable is one of the following statistics:

Variable	Description
proxy.process.arm.num_bypass_on_bad_client_request	Displays the number of times Traffic Server bypassed the cache because it detected non-HTTP traffic on port 80.
proxy.process.arm.num_bypass_on_400	Displays the number of times Traffic Server bypassed the cache because it detected an HTTP 400 error.
proxy.process.arm.num_bypass_on_401	Displays the number of times Traffic Server bypassed the cache because it detected an HTTP 401 error.
proxy.process.arm.num_bypass_on_403	Displays the number of times Traffic Server bypassed the cache because it detected an HTTP 403 error.
proxy.process.arm.num_bypass_on_405	Displays the number of times Traffic Server bypassed the cache because it detected an HTTP 405 error.
proxy.process.arm.num_bypass_on_406	Displays the number of times Traffic Server bypassed the cache because it detected an HTTP 406 error.
proxy.process.arm.num_bypass_on_408	Displays the number of times Traffic Server bypassed the cache because it detected an HTTP 408 error.
proxy.process.arm.num_bypass_on_500	Displays the number of times Traffic Server bypassed the cache because it detected an HTTP 500 error.

Static bypass rules

In addition to adaptively learning what to bypass, Traffic Server allows you to manually configure bypass rules to direct requests from certain clients or to particular origin servers around Traffic Server.

For example, you might want client IP addresses that did not pay for a caching service to be steered around the cache, while paying clients can obtain the benefits of caching. Or you may wish to remove some servers from caching lists, because they do not wish to have their pages cached.

You can configure three types of static bypass rules:

✔ Source bypass, where Traffic Server bypasses a particular source IP address or range of IP addresses. For example, you can use this solution to bypass clients who want to opt out of a caching solution.

✔ Destination bypass, where Traffic Server bypasses a particular destination IP address or range of IP addresses. For example, these could be origin servers who use IP authentication based on the client’s real IP address. Destination bypass rules prevent Traffic Server from caching an entire site. You will experience hit rate impacts if the site you bypass is popular.

- ✔ Source/destination pair bypass, where Traffic Server bypasses requests that originate from the specified source to the specified destination. For example, you could route around specific client-server pairs that experience broken IP authentication or out of band HTTP traffic problems when cached.

  Source/destination bypass rules might be preferable to destination rules because they block a destination server only for those particular users that experience problems.

To configure static bypass rules, edit the `bypass.config` file (refer to *bypass.config, on page 230*).

Viewing the current set of bypass rules

The ARM has a supporting utility called `print_bypass` that allows you to view the current dynamic and static bypass rules.

▼ To view all current dynamic and static bypass rules:

1 In UNIX, log on to a Traffic Server node as the Traffic Server administrator, then make Traffic Server's `bin` directory your working directory.

In Windows, open a Command Prompt window, then `cd` to the `bin` directory (located in the Traffic Server installation directory).

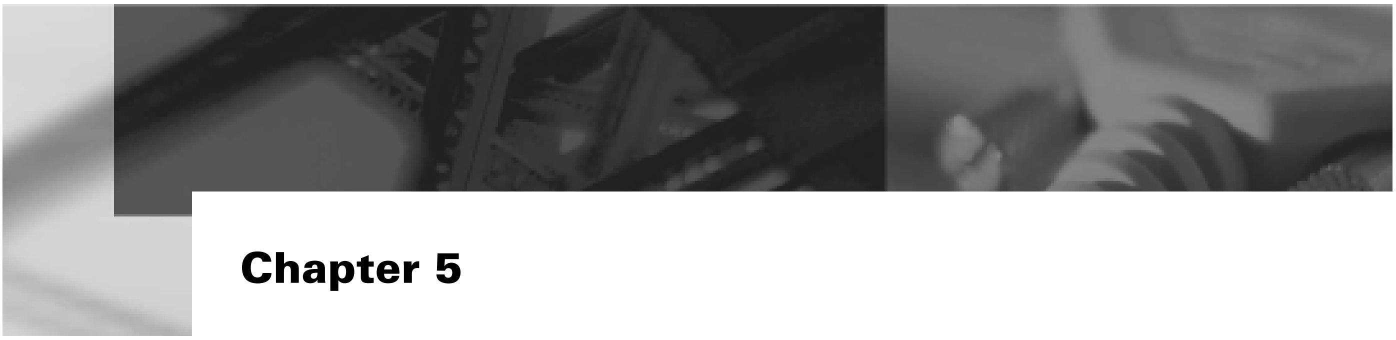
2 Enter the following command at the prompt and press Return:

```
print_bypass
```

All current static and dynamic bypass rules display on screen. The rules are sorted by IP address. You can direct the output of `print_bypass` to a file and save it.

Configuring ARM security

To prevent unauthorized access to machines running Traffic Server, you can configure the ARM to utilize an access control list employing administrator-specified rules to either allow or deny other computers from communicating with the machine. This enables you to effectively create a firewall in front of Traffic Server, thereby denying potentially malicious packets from even reaching the TCP/IP stack on the machine. Refer to *Controlling host access to the Traffic Server machine (ARM security), on page 134*.



Chapter 5

Reverse Proxy and HTTP Redirects

As a reverse proxy cache, Traffic Server serves requests on behalf of origin servers. Traffic Server is configured in such a way that it appears to clients like a normal origin server.

Using HTTP redirects, Traffic Server routes HTTP requests automatically without contacting the origin server.

This chapter discusses the following topics:

- *Understanding reverse proxy caching, on page 66*
- *HTTP Reverse Proxy, on page 68*
- *FTP Reverse Proxy, on page 73*
- *Redirecting HTTP Requests, on page 78*

Understanding reverse proxy caching

In forward proxy caching, Traffic Server handles web requests to distant origin servers on behalf of the clients requesting the content. *Reverse proxy caching* (also known as server acceleration or virtual web hosting) is different in that Traffic Server acts as a proxy cache on behalf of the origin servers that store the content. Traffic Server is configured to be *the* origin server the user is trying to connect to (the origin server's advertised hostname resolves to Traffic Server, which is acting as the real origin server).

Reverse proxy solutions

There are many ways in which Traffic Server can be used as a reverse proxy. Here are a few example scenarios.

You can use Traffic Server in reverse proxy mode to:

- ✔ Off load heavily used origin servers
- ✔ Deliver content efficiently in geographically dispersed areas
- ✔ Provide security for origin servers that contain sensitive information

Off loading heavily used origin servers

Traffic Server can absorb the main origin server request traffic to improve the speed and quality of service of web serving by reducing load and hot spots on backup origin servers.

For example, a web hoster can maintain a scalable Traffic Server serving engine and a set of low-cost, low-performance, less reliable PC origin servers as backup servers. In fact, a single Traffic Server can act as the virtual origin server for multiple backup origin servers, as shown in *Figure 10*.

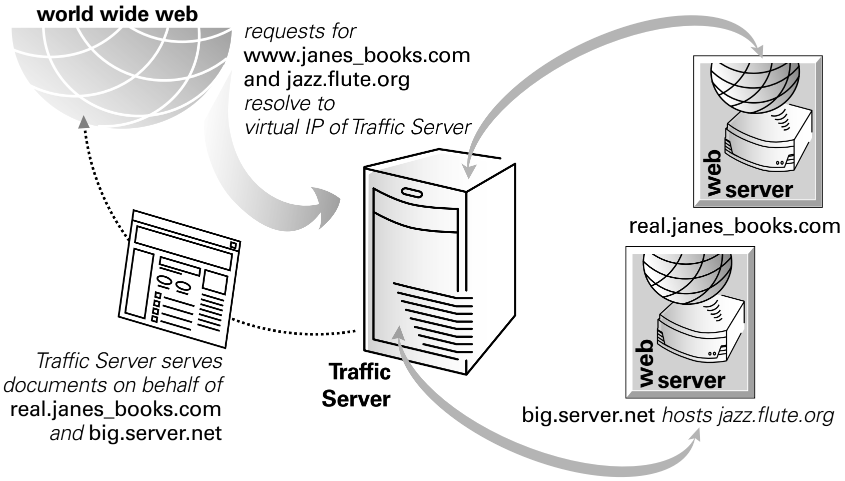


Figure 10 Traffic Server as reverse proxy for a pair of origin servers

Delivering content in geographically dispersed areas

Traffic Server can be used in reverse proxy mode to accelerate origin servers that provide content to geographically dispersed areas. Caches can be easier to manage and more cost-effective than replicating data. For example, Traffic Server can be used as a mirror site on the far side of a trans-Atlantic link to serve users without having to fetch the request and content across expensive international connections. Unlike replication, where hardware must be configured to replicate all data and to handle peak capacity, Traffic Server dynamically adjusts to best utilize the serving and storing capacity of the hardware. Also, Traffic Server is designed to keep content fresh automatically, therefore eliminating the complexity of updating remote origin servers.

Providing security for an origin server

Traffic Server can be used in reverse proxy mode to provide security for an origin server. If you have an origin server that contains sensitive information that you want to keep secure inside your firewall, you can use a Traffic Server outside the firewall as a reverse proxy for that origin server. When outside clients try to access the origin server, their requests go to Traffic Server instead. If the desired content is *not* sensitive, it can be served from the cache. If the content *is* sensitive and not cacheable, Traffic Server obtains the content from the origin server (the firewall allows only Traffic Server access to the origin server). The sensitive content resides on the origin server, safely inside the firewall.

How does reverse proxy caching work?

When a browser makes a request, it normally sends that request directly to the origin server. When Traffic Server is in reverse proxy mode, it must intercept the request for that origin server.

This is done by setting up the DNS entry for the origin server (the origin server's *advertised* hostname) to resolve to the Traffic Server's IP address. When Traffic Server is configured as the origin server, the browser will connect to Traffic Server rather than the origin server.

Note The origin server's hostname and its advertised hostname cannot be the same or there would be a DNS conflict.

The way that Traffic Server receives and processes requests for content in reverse proxy mode differs according to protocol. For information about using and configuring reverse proxy for HTTP requests, refer to *HTTP Reverse Proxy, on page 68*. For information about using and configuring reverse proxy for FTP requests, refer to *FTP Reverse Proxy, on page 73*.

HTTP Reverse Proxy

In forward proxy caching, Traffic Server acts as a proxy server and receives proxy requests. In reverse proxy caching, because Traffic Server is advertised as the origin server, Traffic Server needs to act as an origin server rather than a proxy server, meaning that it receives server requests, not proxy requests. To satisfy proxy requests, Traffic Server must construct a proxy request from the server request.

In HTTP, server requests differ from proxy requests. The main difference is that server requests do not specify the entire URL, just the path. A server request might look like this:

```
GET /index.html HTTP/1.0
```

```
HOST: real.janes_books.com
```

Whereas the corresponding proxy request would look like this:

```
GET http://real.janes_books.com/index.html HTTP/1.0
```

```
HOST: real.janes_books.com
```

Traffic Server can construct a proxy request from a server request by using the server information in the host header.

However, the correct proxy request must contain the hostname of the origin server, not the advertised hostname that the name servers associate to Traffic Server. The advertised hostname is the name that appears in the host header. For example, for the origin server `real.janes_books.com` in *Figure 10, on page 66*, the server request and host header would be:

```
GET /index.html HTTP/1.0
```

```
HOST: www.janes_books.com
```

And the correct proxy request should be:

```
GET http://real.janes_books.com/index.html HTTP/1.0
```

```
HOST: real.janes_books.com
```

To translate `www.janes_books.com` to `real.janes_books.com` Traffic Server needs a set of URL rewriting rules (*mapping rules*). Mapping rules are described in *Using mapping rules, on page 69*.

Generally, you use reverse proxy mode to support more than one origin server. In this case, all of the advertised hostnames resolve to the IP address or virtual IP address of Traffic Server. Using host headers, Traffic Server is able to translate server requests for any number of servers into proxy requests for those servers.

If Traffic Server receives requests from older browsers that do not support host headers, Traffic Server can route these requests directly to a specific server, or send the browser to a URL containing information about the problem. Refer to *Setting HTTP reverse proxy options, on page 71*.

Handling origin server redirect responses

Origin servers often send redirect responses (redirects) back to browsers redirecting them to different pages. For example, if an origin server is overloaded, it might redirect browsers to a less loaded server. Origin servers also redirect when web pages have moved to different locations. When Traffic Server is configured as a reverse proxy, it must readdress redirects from origin servers so that browsers are redirected to Traffic Server, not to another origin server.

To readdress redirects, Traffic Server uses reverse-map rules. In general, you should set up a reverse-map rule for each map rule. To create reverse-map rules, refer to *Using mapping rules* below.

Using mapping rules

Traffic Server uses two types of mapping rules for HTTP reverse proxy:

- ✔ A *map rule* translates the URL in client requests into the URL where the content is located (refer to *Map rules* below)
- ✔ A *reverse-map rule* translates the URL in origin server redirect responses to point to the Traffic Server so that clients are redirected to Traffic Server instead of accessing an origin server directly (refer to *Reverse-map rules* below)

Both map and reverse-map rules consist of a *target* (origin) URL and a *replacement* (destination) URL. In a *map* rule, the target URL points to Traffic Server and the replacement URL specifies where the original content is located. In a *reverse-map* rule, the target URL specifies where the original content is located and the replacement URL points to Traffic Server. Traffic Server stores mapping rules in the `remap.config` file located in Traffic Server's `config` directory.

Map rules

When a Traffic Server in reverse proxy mode receives an HTTP client request, it first constructs a complete request URL from the relative URL and its headers. Traffic Server then compares the complete request URL with its list of target URLs in the `remap.config` file, looking for a match. For the request URL to match a target URL, the following conditions must be true:

- ✔ The scheme of both URLs must be the same
- ✔ The host in both URLs must be the same (if the request URL contains an unqualified hostname, it will never match a target URL with a fully qualified hostname)
- ✔ The ports in both URLs must be the same (if no port is specified in a URL, the default port for the scheme of the URL is used)
- ✔ The path portion of the target URL must match a prefix of the request URL

If Traffic Server finds a match, it translates the request URL into the replacement URL listed in the map rule. It sets the host and path of the request URL to match the replacement URL. If the URL contains path prefixes, Traffic Server removes the prefix of the path that matches the target URL and substitutes it with the path from the replacement URL.

If two mappings match a request URL, Traffic Server applies the first mapping listed in the `remap.config` file.

Reverse-map rules

Reverse-map rules rewrite location headers in origin server responses, instead of client requests. Origin servers use location headers to redirect clients to another location.

For example if there is a directory `/pub` on an origin server at `www.molasses.com`, and a client sends a request to that origin server for `/pub`, the origin server will probably reply with a redirect to `http://www.test.com/pub/` to let the client know that it was a directory it had requested, instead of a document. (A common use of redirects is to normalize URLs so that clients can bookmark documents properly.)

Traffic Server uses reverse-map rules to prevent redirects from origin servers from causing clients to bypass the Traffic Server in favor of direct access to the origin servers.

Setting map and reverse-map rules

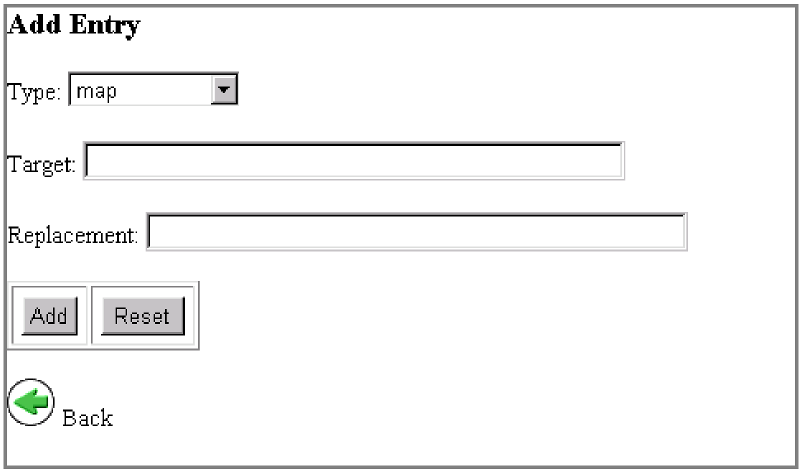
You can set mapping rules by using the Traffic Manager UI or by editing a configuration file manually. Both procedures are described below.

▼ To create a mapping rule from Traffic Manager:

1. From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).
2. On the **Configure** tab, click the **Routing** button.
3. In the **Mapping/Redirection** section of the **Routing** page, click the **Edit Mapping Rules** link.

   The Routing: URL Rewriting page opens.
4. Click the **Add Entry** button.

   The Add Entry page opens (shown below).



Add Entry

Type: map

Target:

Replacement:

5. From the **Type** field, select the type of rule you want to set (`map` or `reverse_map`).
6. In the **Target** field, enter the origin or *from* URL for the rule.
7. In the **Replacement** field, enter the destination or *to* URL for the rule.
8. Click the **Add** button to add the rule.
9. Click the **Make These Changes** button.

▼ To create a mapping rule manually:

1. In a text editor, open the `remap.config` file located in the `config` directory.
2. Enter the mapping rules. Each mapping rule must be on a separate line and must consist of three space-delimited fields in the following format:

```
type target replacement
```

The following table describes the format for each field.

Field	Description
type	Enter either one of the following: `map`—translates an incoming request URL to the appropriate origin server URL. `reverse_map`—translates the URL in origin server redirect responses to point to the Traffic Server.

Field	Description
target	Enter the origin or *from* URL. You can enter up to four components: `scheme://host:port/path_prefix`
replacement	Enter the destination or *to* URL. You can enter up to four components: `scheme://host:port/path_prefix`

The following example shows a map rule that translates all requests for `www.x.com` to the origin server `server.hoster.com`:

```
map http://www.x.com/ http://server.hoster.com
```

For more examples of mapping rules, refer to *remap.config, on page 288*.

3 Save and close the `remap.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

Setting HTTP reverse proxy options

In addition to mapping rules (described in *Using mapping rules, on page 69*), Traffic Server provides several configuration options for reverse proxy that let you:

✔ Enable and disable HTTP reverse proxy mode

✔ Configure Traffic Server to retain the client host header information in a request during translation

✔ Configure Traffic Server to serve requests to the origin servers listed in the mapping rules only (requests from origin servers not listed in the mapping rules are not served)

✔ Specify an alternate URL where incoming requests from older clients that do not provide Host headers are directed

You can set reverse proxy configuration options from the Traffic Manager UI or by editing a configuration file manually. Both procedures are provided below.

▼ To set reverse proxy options from Traffic Manager:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Routing** button.

3 Scroll to the **Reverse Proxy** section of the **Routing** page.

4 Select the **Reverse Proxy: On** button to enable HTTP reverse proxy mode.
Select the **Reverse Proxy: Off** button to disable HTTP reverse proxy mode.

5 Select the **Retain Client Host Header: On** button if you want to retain the client host header in a request (Traffic Server will not translate the client host header).

6 In the **Mapping/Redirection** section, select the **Serve Mapped Hosts Only:On** button if you want Traffic Server to serve requests only from the origin servers listed in the mapping rules. This option provides added security for your Traffic Server system.

7 In the **URL to redirect requests without Host header** field, enter an alternate URL to which incoming requests from older clients that do not provide a host header are directed.

8 Click the **Make These Changes** button.

▼ To set reverse proxy options manually:

1 In a text editor, open the `records.config` file located in the `config` directory.

2 Edit the following variables:

Variable	Description
proxy.config.reverse_proxy.enabled	Set this variable to 1 to enable HTTP reverse proxy mode. Set this variable to 0 (zero) to disable HTTP reverse proxy mode.
proxy.config.url_remap.pristine_host_hdr	Set this variable to 1 to retain the client host header in the request. Set this variable to 0 (zero) if you want Traffic Server to translate the client host header.
proxy.config.url_remap.remap_required	Set this variable to 1 if you want Traffic Server to serve requests only from the origin servers listed in the mapping rules of the remap.config file. Set this variable to 0 (zero) if you want Traffic Server to serve requests from all origin servers.
proxy.config.header.parse.no_host_url_redirect	Enter the URL to which to redirect requests with no host headers.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

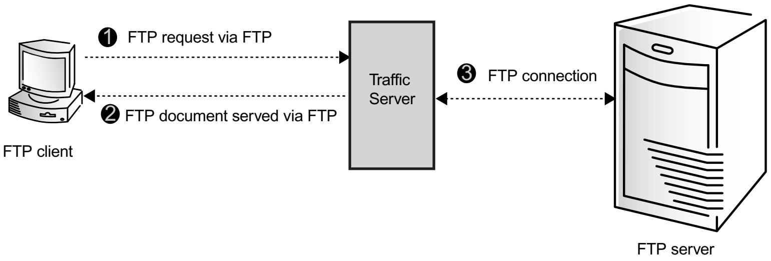
In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

FTP Reverse Proxy

In FTP reverse proxy mode, Traffic Server receives FTP requests from FTP clients on behalf of an FTP server (the FTP server's hostname resolves to Traffic Server's IP address).

Figure 11 illustrates how an FTP request from an FTP client is processed by Traffic Server running in FTP reverse proxy mode.



❶ An FTP client sends a request to an FTP Server via an FTP connection. Because the FTP server's hostname resolves to Traffic Server's IP address, Traffic Server receives the FTP request.

❷ If the request is a cache hit and the content is fresh, Traffic Server sends the requested document to the FTP client via FTP.

❸ If the request is a cache miss or is stale, Traffic Server communicates with the FTP Server via FTP and obtains the requested document. Traffic Server then sends the document to the FTP client via an FTP connection and saves a copy in its cache.

Figure 11 Traffic Server processes an FTP request in FTP reverse proxy mode

Configuring FTP Reverse Proxy

To use FTP reverse proxy, you must:

- ✔ Set FTP mapping rules in the `ftp_remap.config` file, refer to *Setting FTP Mapping Rules, on page 74*
- ✔ Enable the FTP reverse proxy option, refer to *Enabling FTP Reverse Proxy, on page 74*

As an optional configuration step, you can modify FTP options (for example, you can change the FTP connection mode and inactivity timeouts), refer to *Modifying FTP Options, on page 75*.

Setting FTP Mapping Rules

You must set FTP mapping rules so that Traffic Server can direct any incoming FTP requests to the FTP server if the requested document is a cache miss or is stale. You set FTP mapping rules in the `ftp_remap.config` file located in Traffic Manager's `config` directory.

▼ To set FTP mapping rules:

1. In a text editor, open the `ftp_remap.config` file located in Traffic Server's `config` directory.
2. Enter one mapping rule per line in the following format:

   ```
   Traffic_Server_ipaddress:port ftp_Server_ipaddress:port
   ```

   where:

   `Traffic_Server_ipaddress` is the IP address assigned to Traffic Server and *`ftp_Server_ipaddress`* is the IP address assigned to the FTP server to which you want to redirect the FTP requests.

Note Because FTP requests do not include host headers, Traffic Server cannot distinguish between different FTP servers. Therefore, if you are working with multiple FTP servers, you must have multiple IP addresses assigned to Traffic Server.

3. Save and close the `ftp_remap.config` file.
4. In *UNIX*, make Traffic Server's `bin` directory your working directory.

   In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).
5. Run the command `traffic_line -x` to apply the configuration changes.

Enabling FTP Reverse Proxy

You enable the FTP reverse proxy option by editing the `records.config` file. Follow the procedure below.

▼ To enable FTP reverse proxy:

1. In a text editor, open the `records.config` file located in the `config` directory.
2. Edit the following variables:

Variable	Description
proxy.config.ftp.ftp_enabled	Set this variable to 1 to enable FTP on your Traffic Server. This variable must be enabled for Traffic Server to process FTP requests.
proxy.config.ftp.reverse_ftp_enabled	Set this variable to 1 to enable the FTP reverse proxy option. Set this variable to 0 (zero) to disable the FTP reverse proxy option. Note: If this variable is set to 0, but the proxy.config.ftp.ftp_enabled variable (described above) is set to 1, Traffic Server will serve FTP requests in forward proxy mode.

3. Save and close the `records.config` file.
4. In *UNIX*, make Traffic Server's `bin` directory your working directory.

   In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).
5. Run the command `traffic_line -x` to apply the configuration changes.

Modifying FTP Options

After you have set FTP mapping rules and have enabled FTP reverse proxy, Traffic Server can serve FTP requests in reverse proxy mode. Traffic Server uses the default FTP options, such as the data connection mode and connection timeouts, specified in the `records.config` file. You can modify the default FTP options to better suit your needs.

▼ To modify FTP options:

1 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

2 Go to the `FTP Engine` section of the file.

3 Edit the following variables:

Variable	Description
proxy.config.ftp.data_connection_mode	Set this variable to specify the FTP connection mode: 1 = PASV then PORT 2 = PORT only 3 = PASV only
proxy.config.ftp.control_connection_timeout	Set this variable to specify how long Traffic Server waits for a response from the FTP server.
proxy.config.ftp.cache_enabled	Set this variable to 1 to enable FTP document caching for requests sent from an FTP client. Traffic Server will cache the FTP documents it serves. Set this variable to 0 (zero) to disable FTP document caching for requests sent from an FTP client. Traffic Server always obtains the requested FTP document from the FTP server and does not cache it.
proxy.config.ftp.logging_enabled	Set this variable to 1 to enable logging of FTP transactions. Set this variable to 0 (zero) to disable logging of FTP transactions.
proxy.config.ftp.proxy_server_port	Set this variable to specify the port used for FTP connections.
proxy.config.ftp.min_lisn_port	Set this variable to specify the lowest port in the range of listening ports used by Traffic Server for data connections when the FTP client sends a PASV or Traffic Server sends a PORT to the FTP server.
proxy.config.ftp.max_lisn_port	Set this variable to specify the highest port in the range of listening ports used by Traffic Server for data connections when the FTP client sends a PASV or Traffic Server sends a PORT to the FTP server.
proxy.config.ftp.server_data_default_pasv	Set this variable to specify the default method used to set up server side data connections. 1 specifies that Traffic Server sends a PASV to the FTP server and lets the FTP server open a listening port. 0 specifies that Traffic Server is going to try PORT first (setup a listening port on the Traffic Server side of the connection).
proxy.config.ftp.try_pasv_times	Set this variable to specify the number of times Traffic Server can try to open a listening port when the FTP client sends a PASV.

Variable	Description
proxy.config.ftp.try_port_times	Set this variable to specify the maximum number of times Traffic Server can try to open a listening port when sending a PORT to the FTP server.
proxy.config.ftp.try_server_ctrl_connect_times	Set this variable to specify the maximum number of times Traffic Server can try to connect to the FTP server's control listening port.
proxy.config.ftp.try_server_data_connect_times	Set this variable to specify the maximum number of times Traffic Server can try to connect to the FTP server's data listening port when it sends a PASV to the FTP server and gets the ip/listening port information.
proxy.config.ftp.try_client_data_connect_times	Set this variable to specify the maximum number of times Traffic Server can try to connect to the FTP client's data listening port when the client sends a PORT with the ip/listening port information.
proxy.config.ftp.client_ctrl_no_activity_timeout	Set this variable to specify the no activity timeout for the FTP client control connection.
proxy.config.ftp.client_ctrl_active_timeout	Set this variable to specify the active timeout for the FTP client control connection.
proxy.config.ftp.server_ctrl_no_activity_timeout	Set this variable to specify the inactivity timeout for the FTP server control connection.
proxy.config.ftp.server_ctrl_active_timeout	Set this variable to specify the active timeout for the FTP server control connection.
proxy.config.ftp.pasv_accept_timeout	Set this variable to specify the timeout value for a listening data port in traffic server (for PASV, for the FTP client data connection)
proxy.config.ftp.port_accept_timeout	Set this variable to specify the timeout value for a listening data port in traffic server (for PORT, for the FTP server data connection)
proxy.config.ftp.share_ftp_server_ctrl_enabled	Set this variable to 1 to enable sharing of server control connections among multiple anonymous FTP clients. Set this variable to 0 (zero) to disable sharing of server control connections among multiple anonymous FTP clients.
proxy.config.ftp.server_ctrl_keep_alive_no_activity_timeout	Set this variable to specify the timeout value when the FTP server control connection is not used by any FTP clients.
proxy.config.ftp.login_info_fresh_in_cache_time	Set this variable to specify how long the 220/230 responses (login messages) can stay fresh in the cache.
proxy.config.ftp.directory_listing_fresh_in_cache_time	Set this variable to specify how long directory listings can stay fresh in the cache.
proxy.config.ftp.file_fresh_in_cache_time	Set this variable to specify how long FTP files can stay fresh in the cache.

Variable	Description
proxy.config.ftp.simple_directory_listing_cache_enabled	Set this variable to 1 to enable caching of directory listings without arguments (for example, `dir/ls`). Set this variable to 0 (zero) to disable caching of directory listings without arguments (for example, `dir/ls`).
proxy.config.ftp.full_directory_listing_cache_enabled	Set this variable to 1 to enable caching of directory listings with arguments (for example, `ls -al, ls *.txt`). Set this variable to 0 (zero) to disable caching of directory listings with arguments (for example, `ls -al, ls *.txt`).

4 Save and close the `records.config` file.

5 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

6 Run the command `traffic_line -x` to apply the configuration changes.

Redirecting HTTP Requests

You can configure Traffic Server to redirect HTTP requests without having to contact any origin servers. For example, if you redirect all requests for `http://www.ultraseek.com` to `http://www.inktomi.com/products/portal/search/`, all HTTP requests for `www.ultraseek.com` go directly to `www.inktomi.com/products/portal/search.`

You can configure Traffic Server to perform permanent or temporary redirects. Permanent redirects notify the browser of the URL change (by returning an HTTP status code 307) so that the browser can update bookmarks. Temporary redirects notify the browser of the URL change for the current request only (by returning an HTTP status code 301).

▼ To set redirect rules:

1. In a text editor, open the `remap.config` file located in Traffic Server's `config` directory.
2. For each redirect you want to set, enter a mapping rule. Each mapping rule must be on a separate line and must consist of three space-delimited fields: `type`, `target`, and `replacement`. The following table describes the format for each field.

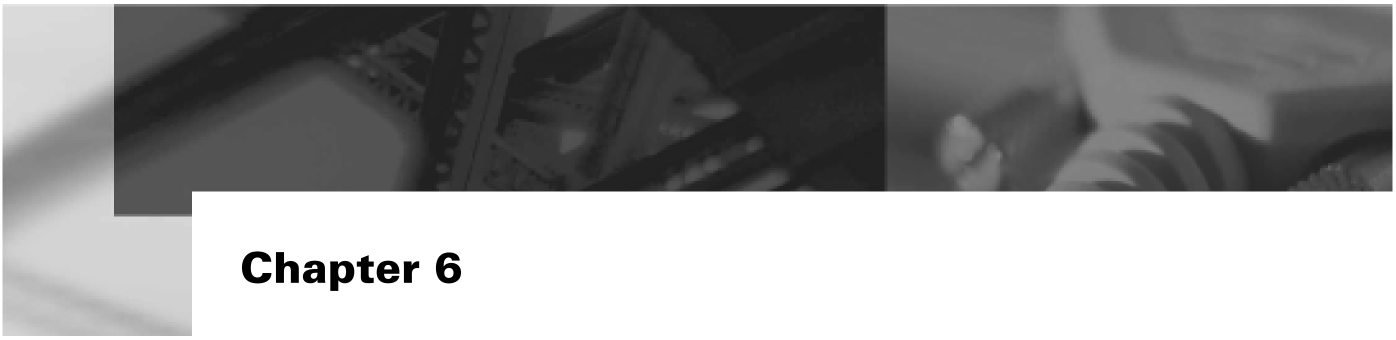
Field	Description
type	Enter either one of the following: `redirect`—redirects HTTP requests permanently without having to contact the origin server `redirect_temporary`—redirects HTTP requests temporarily without having to contact the origin server.
target	Enter the origin or *from* URL. You can enter up to four components: *scheme*`://`*host*`:`*port*`/`*path_prefix*
replacement	Enter the destination or *to* URL. You can enter up to four components: *scheme*`://`*host*`:`*port*`/`*path_prefix*

The following example permanently redirects all HTTP requests for `www.inktomi` to `www.inktomi2.com`.

```
redirect http://www.inktomi.com http://www.inktomi2.com
```

3. Save and close the `remap.config` file.
4. In *UNIX*, make Traffic Server's `bin` directory your working directory.

   In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).
5. Run the command `traffic_line -x` to apply the configuration changes.



Chapter 6

Traffic Server Clusters

Traffic Server scales from a single node to multiple nodes that form a cluster allowing you to improve system performance and reliability.

This chapter discusses the following topics:

Understanding Traffic Server clusters

A Traffic Server cluster consists of multiple Traffic Server nodes. The nodes in a cluster share configuration information and can form a single logical cache.

Traffic Server detects the addition and deletion of nodes in the cluster automatically and can detect when a node is down. When the *Virtual IP failover* feature (described in *Virtual IP failover, on page 84*) is enabled, the live nodes in a cluster can assume a failed node's responsibilities.

Traffic Server has two clustering modes:

- ✔ Management-only mode (refer to *Management-only clustering* below)
- ✔ Full-clustering mode (refer to *Full clustering* below)

Management-only clustering

In management-only clustering mode, Traffic Server cluster nodes share configuration information. You can administer all the nodes at the same time.

Traffic Server uses a multicast management protocol to provide a single system image of your Traffic Server cluster. Information about cluster membership, configuration, and exceptions is shared across all nodes and the `traffic_manager` process automatically propagates configuration changes to all the nodes.

Full clustering

In full-clustering mode, as well as sharing configuration information, a Traffic Server cluster distributes its cache across its nodes into a single, virtual object store, rather than replicating the cache node by node. Traffic Server can provide an enormous aggregate cache size and can maximize cache hit rate by storing objects only once across the entire cluster.

A fully-clustered Traffic Server maps objects to specific nodes in the cluster. When a node receives a request, it checks to see if the request is a hit somewhere in the cluster. If the request is a hit on a different node, the node handling the request fetches the object from the hit node and serves it to the client. Traffic Server uses a proprietary inter-node communication protocol to fetch an object from sibling cluster nodes.

If a node fails or is shut down and removed, Traffic Server removes references to the missing node on all nodes in the cluster. If virtual IP failover (described in *Virtual IP failover, on page 84*) is enabled, requests destined for the missing node are handled by another node.

Changing clustering mode

▼ To change clustering mode:

1 In a text editor, open the `records.config` file located in Traffic Server’s `config` directory.

2 Edit the following variable:

Variable	Description
`proxy.config.cluster.type`	Set this variable to: 1 for full-clustering mode. 2 for management-only mode. 3 for no clustering.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server’s `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

6 Wait several minutes, then run the command `traffic_line -M` to restart the `traffic_manager` process on all the nodes in the cluster.

Adding and deleting nodes in a cluster

You can add a node or delete a node from a Traffic Server cluster at any time. When you add a new node to the cluster, Traffic Server detects it automatically. When you remove a node from the cluster, Traffic Server removes all references to the missing node.

Adding nodes to a cluster

Traffic Server can automatically detect new Traffic Server nodes on your network and add them to the cluster, propagating the latest configuration information to the newcomer. This provides a convenient way to bootstrap new machines.

To connect an additional node to a Traffic Server cluster, you need only install Traffic Server software on the new node, making sure that the cluster name and port assignments are the same as those of the existing cluster. Traffic Server automatically recognizes the new node.

Important The nodes in a cluster must be homogeneous; each node must be the same hardware platform and must run the same version of the same operating system.

▼ To add a node to a cluster:

1 Install the appropriate hardware and connect it to your network. (Consult your hardware documentation for hardware installation instructions.)

2 Install the Traffic Server software using the appropriate procedure for installing a cluster node (refer to the *Traffic Server Installation Guide*). During the installation procedure, make sure that:

   ✗ The cluster name that you assign to the new node is the same as the cluster name for the existing cluster

   ✗ The port assignments for the new node are the same as the port assignments used by the other nodes in the cluster

3 Start the Traffic Server. See *Starting Traffic Server, on page 30*.

If you have an existing Traffic Server installation and you want to add that Traffic Server to the cluster, you do *not* have to re-install the Traffic Server software on the node. Instead, you can edit certain configuration variables on the existing Traffic Server. Follow the procedure below.

▼ To add an existing Traffic Server installation to a cluster:

1 In a text editor, open the `records.config` file located in the `config` directory on the node you want to add to the cluster.

2 Edit the following variables:

Variable	Description
proxy.config.cluster.type	Set this variable to: ▮ 1 for full-clustering mode ▮ 2 for management-only mode Clustering modes are described in *Understanding Traffic Server clusters, on page 80.*
proxy.config.proxy_name	Set this variable to the name of Traffic Server cluster. All nodes in a cluster must use the same name.
proxy.config.cluster.mc_group_addr	Set this variable to specify the multicast address for cluster communications. All nodes in a cluster must use the same multicast address.

Variable	Description
proxy.config.cluster.rsport	Set this variable to specify the reliable service port. The reliable service port is used to send data between the nodes in the cluster. All nodes in a cluster must use the same reliable service port. The default value is 8098.
proxy.config.cluster.mcport	Set this variable to specify the multicast port. The multicast port is used for node identification. All nodes in a cluster must use the same multicast port. The default port number is 8099.
proxy.config.cluster.ethernet_interface	Set this variable to specify the network interface for cluster traffic. All nodes in a cluster must use the same network interface.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -L` to restart the `traffic_manager` process on the local node.

Deleting nodes from a cluster

To delete a node from the Traffic Server cluster, you need to edit a configuration variable on the node you want to delete. Follow the procedure below.

▼ To delete a node from a cluster:

1 Stop Traffic Server on the node you want to delete.

2 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

3 Edit the following variable:

Variable	Description
proxy.config.cluster.type	Set this variable to 3 to turn off clustering.

4 Save and close the `records.config` file.

5 Restart the Traffic Server.

Virtual IP failover

The Traffic Server virtual IP failover feature enables Traffic Server to maintain a pool of virtual IP addresses that it assigns to the nodes in the cluster as necessary. These virtual IP addresses are virtual only in the sense that they are not tied to a specific machine; Traffic Server can assign them to any of its nodes. To the outside world, these virtual IP addresses are *the* addresses of the Traffic Server cluster.

Virtual IP failover assures that if a node in the cluster fails, other nodes can assume the failed node's responsibilities. Traffic Server handles virtual IP failover in the following ways:

- ✔ The `traffic_manager` process maintains cluster communication. Nodes automatically exchange statistics and configuration information through multicast communication. If multicast heartbeats are not received from one of the cluster nodes, the other nodes recognize it as down.
- ✔ The `traffic_manager` process reassigns the IP addresses of the failed node to the remaining operational nodes within approximately 30 seconds, so that service can continue without interruption.
- ✔ The IP addresses are assigned to new network interfaces and the new assignment is broadcast to the local network. The IP reassignment is done through a process called *ARP rebinding*.

What are virtual IP addresses?

Virtual IP addresses are really just IP addresses. They are called virtual addresses because they are not tethered to particular machines and can rotate among nodes in a Traffic Server cluster.

It is common for a single machine to represent multiple IP addresses on the same subnet. This machine would have a primary or real IP address bound to its interface card and also serve many more virtual addresses.

You can set up your user base to use a DNS round robin pointing at virtual IP addresses, as opposed to using the real IP addresses of the traffic server machines.

Because virtual IP addresses are not bound to machines, a Traffic Server cluster can steal addresses from inactive traffic server nodes and distribute those addresses among the remaining live nodes. Using a proprietary Inktomi management protocol, Traffic Server nodes communicate their status with their peers. If a node fails, its peers notice the failure and quickly negotiate which of the remaining nodes will mask the fault by taking over the failed node's virtual interface.

Setting virtual IP address options

Traffic Server provides several configuration options for virtual IP addressing. You can:

- ✔ Enable and disable virtual IP addressing (see *Enabling/disabling virtual IP addressing, on page 85*)
- ✔ Add, modify, and delete virtual IP addresses (see *Adding and editing virtual IP addresses, on page 86*)

Enabling/disabling virtual IP addressing

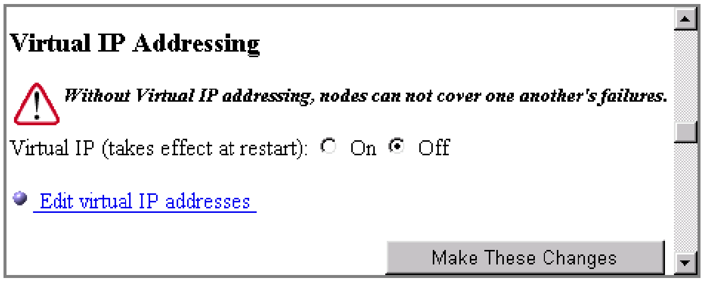
You can turn virtual IP addressing on or off by using the Traffic Manager UI or by editing a configuration file manually. Both procedures are described below.

▼ To enable/disable virtual IP addressing from Traffic Manager:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Server** button.

3 Scroll to the **Virtual IP Addressing** section of the **Server Basics** page (shown below).



Virtual IP Addressing

Without Virtual IP addressing, nodes can not cover one another's failures.

Virtual IP (takes effect at restart): ○ On ● Off

Edit virtual IP addresses

4 Select **Virtual IP: On** to enable virtual IP addressing.
Select **Virtual IP: Off** to disable virtual IP addressing.

5 Click the **Make These Changes** button.

6 Scroll to the **Web Management** section of the **Server Basics** page and click the **restart** button to restart the `traffic_manager` process on all the nodes in the cluster.

▼ To enable/disable virtual IP addressing manually:

1 In a text editor, open the `records.config` file located in the `config` directory.

2 Edit the following variable:

Variable	Description
proxy.config.vmap.enabled	Set this variable to 1 to enable virtual IP addressing. Set this variable to 0 (zero) to disable virtual IP addressing.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

6 Wait several minutes, then run the command `traffic_line -M` to restart the `traffic_manager` process on all the nodes in the cluster.

Adding and editing virtual IP addresses

You can add new or edit existing virtual IP addresses from the Traffic Manager UI or by editing a configuration file manually.

Caution Incorrect IP addressing can effectively disable your system. Make sure you understand how virtual IP addresses work before changing them.

Virtual IP addresses must be pre-reserved like all IP addresses before they can be assigned to Traffic Server.

▼ To add or edit virtual IP addresses from Traffic Manager:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Server** button.

3 Scroll to the **Virtual IP Addressing** section of the **Server Basics** page.

4 Click the **Edit virtual IP addresses** link.

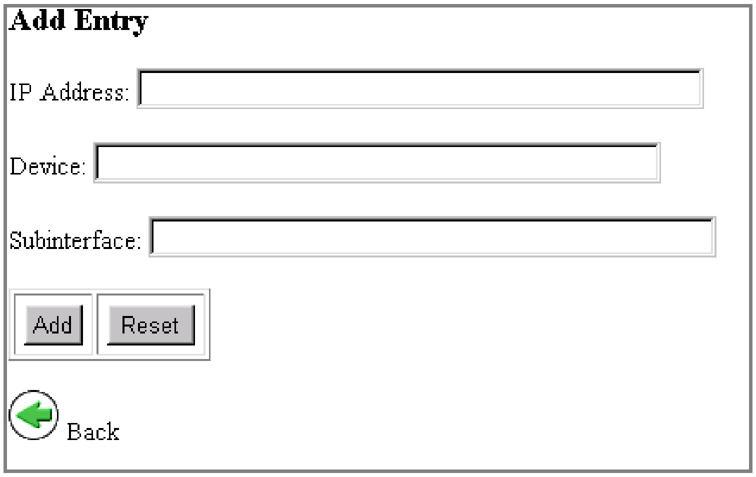
The **Virtual IP** page opens and lists the pool of IP addresses that are assigned to the cluster.

5 To remove or modify a virtual IP address in the list, click the **Delete** or **Modify** button.

6 To add a virtual IP address, click the **Add Entry** button.

The Add Entry page opens (shown below).

Note In Windows, the Add Entry page differs from the one shown below. The **Device** field is not provided and the **Subinterface** field is replaced by the **Interface Name** field.



Add Entry

IP Address:

Device:

Subinterface:

7 In the **IP Address** field, enter the virtual IP address.

8 In the **Device** field (UNIX only), enter the network interface name (for example, `hme0` on Solaris or `tu0` on Digital UNIX).

9 In the **Subinterface** field (*UNIX*), enter the subinterface-ID (this is the number between 1 and 255 that the interface uses for the address).

In the **Interface Name** field (*Windows*), enter the name of the network interface to which you want to assign the virtual IP address.

In Windows, a list of network interfaces and their names is stored in the `winnt_intr.config` file in Traffic Server's `config` directory. The Traffic Server installation program automatically creates an interface name for each available network interface on the system and records it in the winnt_intr.config file. If you are not sure what to enter in the **Interface Name** field, check the winnt_intr.config file.

10 Click the **Add** button.

11 Click the **Make these Changes** button.

12 Scroll to the **Web Management** section of the **Server Basics** page and click the **restart** button to restart the traffic_manager process on all the nodes in the cluster.

▼ To add or edit virtual IP addresses manually:

1 In a text editor, open the vaddrs.config file located in the config directory.

2 To delete or modify existing virtual IP addresses, delete or edit the line that contains the IP address.

3 To add new virtual IP addresses:
In *Unix*, enter one virtual IP address per line using the following format:

UNIX

```
IP address    device   sub interface
```

where: *IP address* is the virtual IP address, *device* is the network interface name (for example, hme0 on Solaris, or tu0 on Digital UNIX), *sub interface* is the subinterface ID (this is a number between 1 and 255 that the interface uses for the address).

In *Windows*, enter one virtual IP address per line using the following format:

Windows

```
IP address    interface
```

where: *IP address* is the virtual IP address and *interface* is the name of the network interface to which you want to assign the virtual IP address.

4 Save and close the vaddrs.config file.

5 In *UNIX*, make Traffic Server's bin directory your working directory.

In Windows, open a command prompt window and change to the bin directory (located in the Traffic Server installation directory).

6 Run the command traffic_line -x to apply the configuration changes.

7 Wait several minutes, then run the command traffic_line -M to restart the traffic_manager process on all the nodes in the cluster.

Chapter 7

Hierarchical Caching

Traffic Server can participate in cache hierarchies, where requests not fulfilled in one cache can be routed to other regional caches, taking advantage of the contents and proximity of nearby caches.

This chapter discusses the following topics.

Understanding cache hierarchies

A cache hierarchy consists of levels of caches that communicate with each another. Traffic Server supports several types of cache hierarchies. All cache hierarchies recognize the concept of parent and child. A parent cache is a cache higher up in the hierarchy, to which Traffic Server can forward requests. A child cache is a cache for which Traffic Server is a parent.

Traffic Server can be a member of the following cache hierarchies:

- ✔ An HTTP cache hierarchy (described in *HTTP cache hierarchies, on page 91*)
- ✔ An ICP (Internet Cache Protocol) cache hierarchy (described in *ICP cache hierarchies, on page 95*)

HTTP cache hierarchies

In an HTTP cache hierarchy, if a Traffic Server node cannot find a requested object in its cache, it can search a parent cache—which itself can search other caches—before resorting to retrieving the object from the origin server.

You can configure a Traffic Server node to use one or more HTTP parent caches. You use more than one HTTP parent cache so that if one parent is unavailable, another parent can service requests. This is called *parent failover* and is described in *Parent failover* below.

Note If you do not want all requests to go to the parent cache, you can configure Traffic Server to route certain requests directly to the origin server (for example, requests that contain specific URLs) by setting parent proxy rules in the `parent.config` configuration file (described in *parent.config, on page 255*).

Figure 12 illustrates a simple cache hierarchy, where a Traffic Server node is configured to use a parent cache.

In this figure, a client sends a request to a Traffic Server node (which is a child in the cache hierarchy because it is configured to forward missed requests to a parent cache). The request is a cache miss, so the Traffic Server forwards the request to the parent cache. On the parent, the request is a cache hit, so the parent sends a copy of the content to the Traffic Server, where it is cached and then served to the client. (Future requests for this content can now be served directly from the Traffic Server cache.)

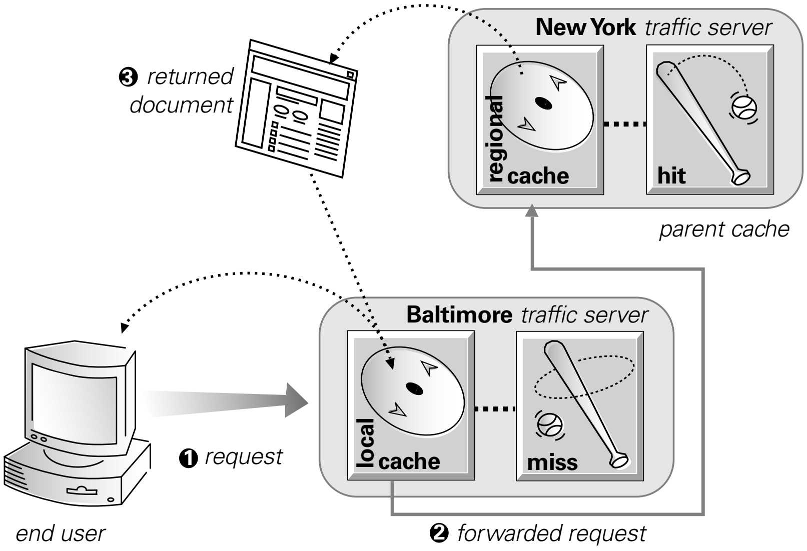


Figure 12 An HTTP cache hierarchy in action

Note If the request is a cache miss on the parent, the parent retrieves the content from the origin server (or from another cache depending on the parent's configuration). The parent caches the content, then sends a copy to the Traffic Server (its child), where it is cached and served to the client.

Parent failover

Traffic Server supports the use of several parent caches so that if one parent cache is not available, another parent cache can service client requests.

When you configure your Traffic Server to use more than one parent cache, Traffic Server detects when a parent is not available and sends missed requests to another parent cache. If you specify more than two parent caches, the order in which the parent caches are queried depends upon the parent proxy rules configured in the parent configuration file described in *parent.config, on page 255*. By default, the parent caches are queried in the order in which they are listed in the configuration file.

Configuring Traffic Server to use an HTTP parent cache

To configure Traffic Server to use one or more parent caches, you must:

- ✔ Enable the HTTP parent caching option (described in *Enabling the HTTP parent caching option* below)
- ✔ Identify the HTTP parent cache(s) you want to use to service missed requests (described in *Identifying HTTP parent caches, on page 94*)

Enabling the HTTP parent caching option

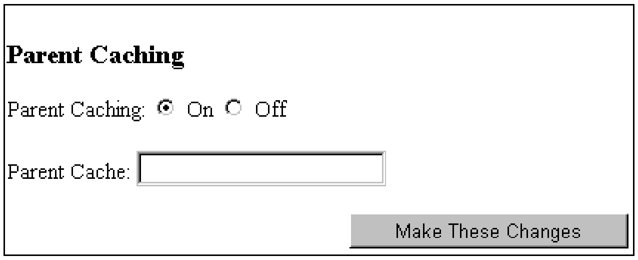
You can enable the HTTP parent caching option by using the Traffic Manager UI or by editing a configuration file manually. Both procedures are described below.

▼ To enable the HTTP parent caching option from Traffic Manager:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Routing** button.

3 Scroll to the **Parent Caching** section of the **Routing** page (shown below).



Parent Caching

Parent Caching: ● On ○ Off

Parent Cache:

4 Select **Parent Caching: On**.

5 Click the **Make These Changes** button.

▼ To enable HTTP parent caching manually:

1 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

2 Edit the following variable:

Variable	Description
proxy.config.http.parent_proxy_routing_enable	Set this variable to 1 to enable the HTTP parent caching option.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

Identifying HTTP parent caches

You must identify the parent to which requests are sent when the Traffic Server cannot find the requested object in its cache. To use *parent failover*, you must identify more than one parent cache so that when a parent cache is unavailable, requests are sent to another parent cache.

You can identify parent caches by using the Traffic Manager UI or by setting proxy rules in the parent configuration file (`parent.config`). To set parent proxy rules, refer to *parent.config, on page 255*.

▼ To identify an HTTP parent cache from Traffic Manager:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Routing** button.

3 Scroll to the **Parent Caching** section of the **Routing** page.

4 Make sure that **Parent Caching: On** is selected.

5 In the **Parent Cache** field, enter the host name and host port of the parent cache to which you want missed requests to be forwarded. Use the following format:

```
parent_name:port_number
```

To specify more than one parent cache for parent failover, end each entry in the **Parent Cache** field with a semi-colon (;). For example:

```
parent_name:port_number; parent_name:port_number;
```

Note When you use the Traffic Manager UI to identify parent caches for parent failover, Traffic Server sends requests to the parents in the order that they appear in the **Parent Cache** field. For example, when the first parent cache listed in the **Parent Cache** field is not available, Traffic Server sends requests to the next parent cache in the list. If all parent caches are unavailable, requests are sent directly to the origin server. You can change the order in which parent caches are queried by setting up a parent proxy rule in the `parent.config` file using the `round_robin` action. Refer to *parent.config, on page 255*.

6 Click the **Make These Changes** button.

ICP cache hierarchies

The Internet Cache Protocol (ICP) is a protocol used by proxy caches to exchange information about their content. ICP query messages ask other caches if they are storing a particular URL. ICP response messages reply with a hit or miss answer.

A cache exchanges ICP messages only with specific ICP peers, which are neighboring caches that can receive ICP messages. An ICP peer can be a sibling cache, which is at the same level in the hierarchy, or a parent cache, which is one level up in the hierarchy.

If Traffic Server has ICP caching enabled, it sends out ICP queries to its sibling caches in the event of a cache miss on an HTTP request. If there are no hits on siblings, Traffic Server sends ICP queries to ICP parents. If there are no hits on ICP parents, Traffic Server forwards the request to its HTTP parents. If there are no HTTP parent caches established, Traffic Server forwards the request to a selected ICP parent cache (which resolves the request by communicating with the origin server).

Note If Traffic Server receives a hit message from an ICP peer, Traffic Server sends the HTTP request to that peer. However, it may be a cache miss, because the original HTTP request contains header information that is not communicated by the ICP query. For example, the hit might not be the requested alternate. If an ICP hit turns out to be a miss, Traffic Server forwards the request to either its HTTP parent caches or to the origin server.

Configuring Traffic Server to use an ICP cache hierarchy

When you configure a Traffic Server node to be part of an ICP cache hierarchy, you must:

✔ Enable ICP caching and set options to:

  ✗ Determine if the Traffic Server can receive ICP messages only or both send and receive ICP messages

  ✗ Determine if Traffic Server can send messages directly to each ICP peer or send a single message on a specified multicast channel

  ✗ Specify the port used for ICP messages

  ✗ Set the ICP query timeout

✔ Identify the ICP peers with which Traffic Server can communicate

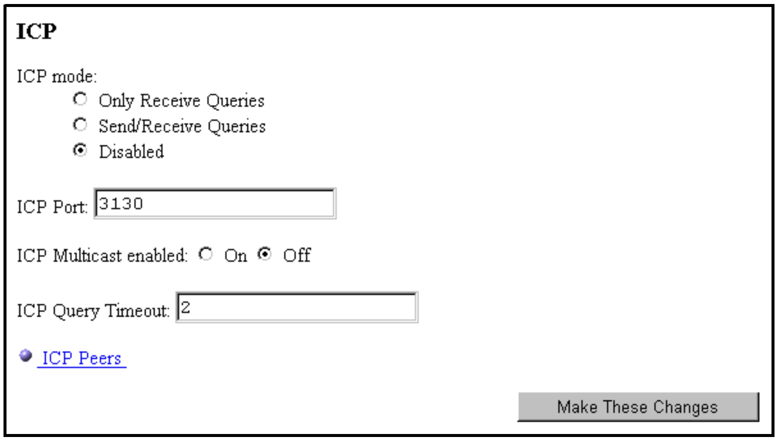
You can set ICP options and identify ICP peers by using the Traffic Manager UI or by editing a configuration file manually. Both procedures are provided below.

▼ To set ICP options from Traffic Manager:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Routing** button.

3 Scroll to the **ICP** section of the **Routing** page (shown below)



ICP

ICP mode:

○ Only Receive Queries
○ Send/Receive Queries
● Disabled

ICP Port: 3130

ICP Multicast enabled: ○ On ● Off

ICP Query Timeout: 2

4 In the **ICP mode** area, select:

- **Only Receive Queries** to configure Traffic Server to receive ICP queries from other ICP peers only. In this mode, Traffic Server cannot send queries to other ICP peers.
- **Send/Receive Queries** to configure Traffic Server to both send and receive ICP queries.
- **Disabled** to turn off ICP hierarchical caching.

5 In the **ICP Port** field, enter the port that you want to use for ICP messages. The default is 3130.

6 Select **ICP Multicast enabled:On** to send ICP messages through multicast if your Traffic Server has a multicast channel connection to its ICP peers.

7 In the **ICP Query Timeout** field, enter the timeout for ICP queries. The default is 2 seconds.

8 Click the **Make These Changes** button.

▼ To set ICP options manually:

1 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

2 Edit the following variables:

Variable	Description
proxy.config.icp.enabled	Set this variable to: ▪ 0 to disable ICP. ▪ 1 to allow Traffic Server to receive ICP queries only. ▪ 2 to allow Traffic Server to send and receive ICP queries.
proxy.config.icp.icp_port	Set this variable to specify the UDP port that you want to use for ICP messages. The default is 3130.
proxy.config.icp.multicast_enabled	Set this variable to: ▪ 0 to disable ICP multicast. ▪ 1 to enable ICP multicast.
proxy.config.icp.query_timeout	Set this variable to specify the timeout used for ICP queries. The default is 2 seconds.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

Identifying ICP Peers

For ICP to work, the Traffic Server must recognize its ICP peers (siblings and parents). You can identify ICP peers by using the Traffic Manager UI or by editing a configuration file manually.

▼ To identify an ICP peer from Traffic Manager:

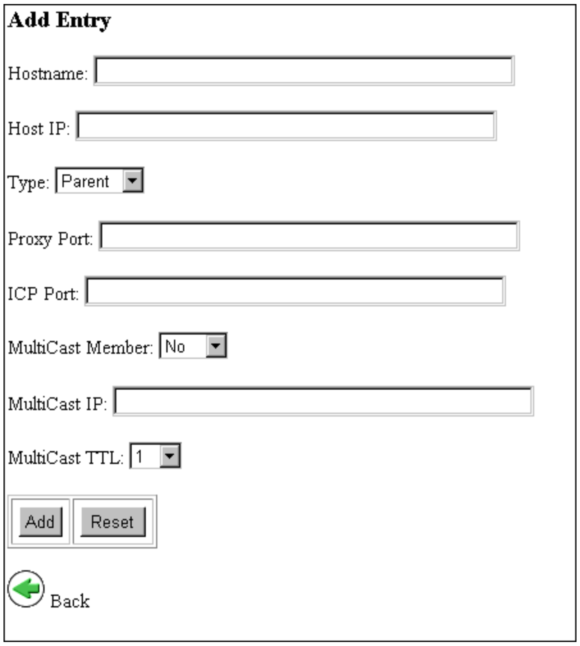
1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Routing** button.

3 Scroll to the **ICP** section of the **Routing** page.

4 Click the **ICP Peers** link.

The **Configure: ICP Peers** page opens (shown below).



Add Entry

Hostname:

Host IP:

Type: Parent

Proxy Port:

ICP Port:

MultiCast Member: No

MultiCast IP:

MultiCast TTL: 1

5 In the **Hostname** field, enter the host name of the ICP peer. The host name is required only if you do not specify the IP address in the **Host IP** field described below.

6 In the **Host IP** field, enter the IP address of the ICP peer.

Note If you do not know the IP address, you can enter 0.0.0.0. Traffic Server uses the host name specified in the **Hostname** field to obtain the IP address via a DNS lookup.

7 From the **Type** drop-down list, select:

✗ **Parent** to indicate that the ICP peer is a parent cache.

✗ **Sibling** to indicate that the ICP peer is a sibling cache.

8 In the **Proxy Port** field, enter the TCP port used by the ICP peer for ICP communication. This is the Traffic Server's proxy port (usually 8080).

9 In the **ICP Port** field, enter the UDP port used by the ICP peer for ICP communication (usually 3130).

10 From the **MultiCast Member** drop-down list, select:

✗ **No** if the ICP peer is *not* on a multicast network with the Traffic Server.

✗ **Yes** if the ICP peer *is* on a multicast network with the Traffic Server.

11 In the **MultiCast IP** field, enter the multicasr IP address.

12 From the **MultiCast TTL** drop-down list, select:

- **1** if you do *not* want IP multicast datagrams to be forwarded beyond a single subnetwork.
- **2** to allow delivery of IP multicast datagrams to more than one subnet (if there are one or more multicast routers attached to the first hop subnet).

13 Click the **Add button**.

14 Click the **Make These Changes** button.

▼ To identify an ICP peer manually:

1 In a text editor, open the `icp.config` file located in Traffic Server's `config` directory.

2 For each ICP peer you want to identify, enter a separate line in the configuration file in the following format:

```
host:host_IP:cache_type:proxy port:icp port:MC_on:MC_IP:MC_TTL:
```

where:

Field	Description
host	Specifies the host name of the ICP peer.
host_IP	Specifies the IP address of the ICP peer.
cache_type	Specifies the type of ICP peer. Enter one of the following: `1` to indicate an ICP parent cache `2` to indicate an ICP sibling cache
proxy_port	Specifies the TCP port number used by the ICP peer for proxy communication. The default is 8080.
icp_port	Specifies the UDP port number used by the ICP peer for ICP communication. The default is 3130.
MC_on	Specifies multicast options. Enter one of the following: `0` if the ICP peer is *not* on a multicast network with the Traffic Server. `1` if the ICP peer is on a multicast network with the Traffic Server.
MC_IP	Specifies the multicast IP address.
MC_TTL	Specifies one of the following options: `1` if you do *not* want IP multicast datagrams to be forwarded beyond a single subnetwork. `2` to allow delivery of IP multicast datagrams to more than one subnet (if there are one or more multicast routers attached to the first hop subnet).

3 Save and close the `icp.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

Chapter 8

Configuring the cache

The Traffic Server cache consists of a high speed object database called the object store that indexes objects according to URLs and associated headers.

This chapter discusses the following topics:

The Traffic Server Cache

The Traffic Server cache consists of a high speed object database called the *object store*. The object store indexes objects according to URLs and their associated headers enabling Traffic Server to store, retrieve, and serve not only web pages, but parts of web pages, providing optimum bandwidth savings. Using sophisticated object management, the object store can cache alternate versions of the same object, varying on spoken language or browser type and can efficiently store very small and very large documents, minimizing wasted space. When the cache begins to fill, the Traffic Server mobilizes *garbage collectors* to remove stale data, ensuring that the most requested objects are kept on-hand and fresh.

Traffic Server is designed to tolerate total disk failures on any of the cache disks. If the disk fails completely, Traffic Server marks the entire disk as corrupt and continues using the remaining disks. An alarm is sent to the Traffic Manager indicating which disk failed. If all of the cache disks fail, Traffic Server goes into proxy-only mode.

You can perform the following cache configuration tasks:

- ✔ Change the total amount of disk space allocated to the cache. Refer to *Changing cache capacity, on page 103*.
- ✔ Partition the cache by reserving cache disk space for specific protocols and origin servers/ domains. Refer to *Partitioning the cache, on page 104*
- ✔ Delete all data in the cache. Refer to *Clearing the cache, on page 107*.

RAM cache

Traffic Server maintains a small RAM memory cache of extremely popular objects. This RAM cache serves the most popular objects as fast as possible and reduces load on disks, especially during temporary traffic peaks. You can configure the RAM cache size to suit your needs. See *Changing the size of the RAM cache, on page 108*.

Changing cache capacity

You can change the total amount of disk space allocated to the cache by either increasing or reducing cache capacity.

Adding cache capacity

You can increase the total amount of disk space allocated to the cache on existing disks or add new disks to a Traffic Server node.

▼ To add cache capacity:

1. Stop Traffic Server.
2. Add hardware, if necessary.
3. Edit the `storage.config` file to increase the amount of disk space allocated to the cache on existing disks or to describe the new hardware you are adding. Refer to *storage.config, on page 296*.
4. Restart Traffic Server.

Reducing cache capacity

You can reduce the total amount of disk space allocated to the cache on an existing disk or remove disks from a Traffic Server node.

▼ To reduce cache capacity:

1. Stop Traffic Server.
2. Remove hardware, if necessary.
3. Edit the `storage.config` file to reduce the amount of disk space allocated to the cache on existing disks or to delete reference to the hardware you are removing. Refer to *storage.config, on page 296*.
4. Restart Traffic Server.

Partitioning the cache

You can manage your cache space more efficiently and restrict disk usage by creating cache partitions of different sizes for specific protocols. You can further configure these partitions to store data from specific origin servers and/or domains.

Important The partition configuration must be the same on all nodes in a cluster.

Creating cache partitions for specific protocols

You can create separate partitions for your cache that vary in size to store content according to protocol. This configuration ensures that a certain amount of disk space is always available for a particular protocol.

▼ To partition the cache according to protocol:

1 Stop Traffic Server.

2 In a text editor, open the `partition.config` file located in Traffic Server's `config` directory.

3 For each partition you want to create, enter a line with the following format:

```
partition=partition_number  scheme=protocol_type  size=partition_size
```

where:

`partition_number` is a number between 1 and 255 (the maximum nuber of partitions is 255).

`protocol_type` is either `http` or `mixt` (all streaming media content is stored in the `mixt` partition and all HTTP, FTP, and NNTP content is stored in the `http` partition).

`partition_size` is the amount of cache space allocated to the partition. This value can be either a percentage of the total cache space or an absolute value. The absolute value must be a multiple of 128 MB, where 128 MB is the smallest value. If you specify a percentage, the size is rounded down to the closest multiple of 128 MB. Each partition is striped across several disks to achieve parallel I/O. For example, if there are 4 disks, a 1 GB partition will have 256 MB on each disk (assuming each disk has enough free space available).

Note If you do not allocate all the disk space in the cache, the extra disk space is not used. You can use the extra space at a later time to create new partitions without deleting and clearing the existing partitions.

The following example partitions the cache evenly between HTTP and streaming media requests:

```
partition=1 scheme=http size=50%
partition=2 scheme=mixt size=50%
```

4 Save and close the `partition.config` file.

5 Restart Traffic Server.

Making changes to partition sizes and protocols

After you have configured your cache partitions based on protocol, you can make changes to the configuration at any time. Before making changes, note the following:

- ✔ You must stop Traffic Server before you change the cache partition size and protocol assignment.
- ✔ When you increase the size of a partition, the contents of the partition are *not* deleted However, when you reduce the size of a partition, the contents of the partition *are* deleted.
- ✔ When you change the partition number, the partition is deleted and then recreated even if the size and protocol type remain the same.
- ✔ When you add new disks to your Traffic Server node, the partition sizes specified in percentages increase proportionately.
- ✔ A lot of changes to the partition sizes may result in disk fragmentation, which affects performance and hit rate. Inktomi recommends that you clear the cache (refer to *Clearing the cache, on page 107*) before making many changes to cache partition sizes.

Partitioning the cache according to origin server or domain

After you have partitioned the cache according to size and protocol, you can assign the partitions you created to specific origin servers and/or domains.

You can assign a partition to a single origin server or multiple origin servers. However, if a partition is assigned to multiple origin servers, there is no guarantee on the space available in the partition for each origin server. Content is stored in the partition according to popularity.

In addition to assigning partitions to specific origin servers and domains, you must assign a generic partition to store content from all origin servers and domains that are not listed. This generic partition is also used if the all partitions for a particular origin server or domain become corrupt.

Important If you do not assign a generic partition, you will be unable to start Traffic Server.

Note You do *not* need to stop Traffic Server before you assign partitions to particular hosts or domains. However, this type of configuration can cause a spike in memory usage and is time consuming. Inktomi recommends that you configure partition assignment during periods of low traffic.

▼ To partition the cache according to hostname and domain:

1. Configure the cache partitions according to size and protocol as described in *Creating cache partitions for specific protocols, on page 104*.

   You should create a separate partition based on protocol for each host and domain, and an additional generic partition to use for content that does not belong to these origin servers or domains. For example, if you want to separate content from two different origin servers, and you want the content to be separated by protocol so that HTTP content and streaming media content is stored separately, you must have five separate partitions. One HTTP-based partition for each origin server, one streaming media-based partition for each origin server, and a generic partition for all other origin servers not listed (the partitions do not have to be the same size).
2. In a text editor, open the `hosting.config` file located in Traffic Server's `config` directory.
3. Enter a line in the file to allocate the partition(s) used for each origin server and/or domain. For an origin server, the line must contain the following format:

```
hostname=hostname  partition=list_of_partition_numbers
```

For a domain, the line must contain the following format:

```
domain=domain_name  partition=list_of_partition_numbers
```

where:

hostname is the fully qualified hostname of the origin server whose content you want to store on a particular partition (for example, `www.inktomi.com`).

domain_name is the domain whose content you want to store on a particular partition (for example, `inktomi.com`).

list_of_partition_numbers is a comma-separated list of the partitions on which you want to store the content that belongs to the origin servers or domains listed. The partition numbers must be valid numbers listed in the `partition.config` file.

In the following example, content from the domain `inktomi.com` is stored on partition 1 and 2, while content from `www.yahoo.com` is stored on partition 3.

```
domain=inktomi.com partition=1,2
hostname=www.yahoo.com partition=3
```

Note If you want to allocate more than one partition to an origin server or domain, you must enter the partitions in a comma-separated list on one line as shown in the above example. The file cannot contain multiple entries for the same origin server or domain.

4 Assign a generic partition to use for content that does not belong to any of the origin servers or domains listed in the file. If all partitions for a particular origin server become corrupt, Traffic Server will also use the generic partition to store content for that origin server.

Enter the following line in the `hosting.config` file

```
hostname=* partition=list_of_partition_numbers
```

where *list_of_partition_numbers* is a comma-separated list of the partitions on which you want to store the content that belongs to all other origin servers and domains not listed in the file.

5 Save and close the `hosting.config` file.

6 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

7 Run the command `traffic_line -x` to apply the configuration changes.

Clearing the cache

When you clear the cache, you remove all data from the entire cache. You should clear the cache before performing certain cache configuration tasks, such as partitioning.

Important Save your current Traffic Server configuration before you clear the cache by taking a snapshot of your configuration (refer to *The Snapshots button, on page 127*).

▼ To clear the cache:

1. Stop the Traffic Server.
2. If Traffic Server's `bin` directory is not in your path, make it your working directory.
3. Enter the following Traffic Server command and press Return.

```
traffic_server -Cclear
```

Caution The `clear` command deletes all data in the Traffic Server cache. You are *not* prompted to confirm the deletion.

4. Enter the following command to restart Traffic Server.

```
start_traffic_server
```

Changing the size of the RAM cache

The Traffic Server provides a dedicated RAM cache for fast retrieval of popular small objects. The default RAM cache size is automatically calculated based on the number and size of the cache partitions you have configured. You can increase the RAM cache size for better cache hit performance.

Caution If you increase the size of the RAM cache and observe a decrease in Traffic Server performance (such as increased latencies), the operating system might require more memory for network resources. Return the RAM cache size to its previous value.

▼ To change RAM cache size:

1 Stop Traffic Server.

2 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

3 Edit the following variable:

Variable	Description
proxy.config.cache.ram_cache.size	Set this variable to specify the size of the RAM cache.

If you have partitioned your cache according to protocol and /or hosts, the size of the RAM cache for each partition is proportional to the size of that partition.

4 Save and close the `records.config` file.

5 Restart Traffic Server.

Chapter 9

Monitoring Traffic

Traffic Server provides several options for monitoring system performance and analyzing network traffic.

This chapter discusses the following topics:

Traffic Server monitoring tools

Traffic Server provides the following tools to monitor system performance and analyze network traffic:

- ✔ The Traffic Manager UI provides statistics that show Traffic Server performance and network traffic information. Refer to *Viewing statistics from Traffic Manager, on page 111*.
- ✔ The Traffic Manager UI presents alarms that signal any detected failure conditions. Refer to *Working with Traffic Manager Alarms, on page 115*.
- ✔ The Traffic Line command-line interface provides an alternative method of viewing Traffic Server performance and network traffic information. The statistics are the same as those you see from the Traffic Manager UI. Refer to *Viewing Statistics from Traffic Line, on page 117*.
- ✔ The MRTG (Multi Router Traffic Grapher) tool provides a variety of graphs that show Traffic Server performance and network traffic information. Refer to *Using MRTG, on page 119*.
- ✔ SNMP (Simple Network Management Protocol) support lets you monitor and manage Traffic Server through SNMP network management facilities. Refer to *Using SNMP, on page 121*.

Viewing statistics from Traffic Manager

You can use the Traffic Manager UI to collect and interpret statistics about Traffic Server performance and web traffic. You view statistics using Traffic Manager's Monitor mode.

Starting Traffic Manager Monitor mode

▼ To start Traffic Manager Monitor mode:

1 Open your web browser.

The Traffic Manager requires Java and JavaScript; be sure to enable Java and JavaScript in your browser.

2 Type one of the following locations in your browser:

Standard

```
http://nodename:adminport/
```

SSL

```
https://nodename:adminport/
```

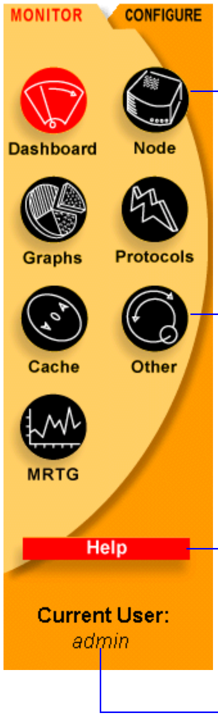
where `nodename` is the name of the Traffic Server node and `adminport` is the number assigned to the Traffic Manager port.

Note Use the SSL `https` command to reach Traffic Manager only if you have restricted access to Traffic Manager via SSL connections; otherwise, use the standard `http` command.

3 If necessary, log on to Traffic Server with the administrator ID and password or your administrator account.

Note The administrator ID and password are set during Traffic Serve installation. You can change the ID and password, as well as create and modify administrator accounts. For more information, refer to *Controlling access to the Traffic Manager UI, on page 136*.

The Traffic Manager displays the **Monitor** tab (shown below).



Click a button to display a page listing statistics you can view

Click this button to display a description of the statistics

Shows the current user logged on to Traffic Manager

Using Monitor mode

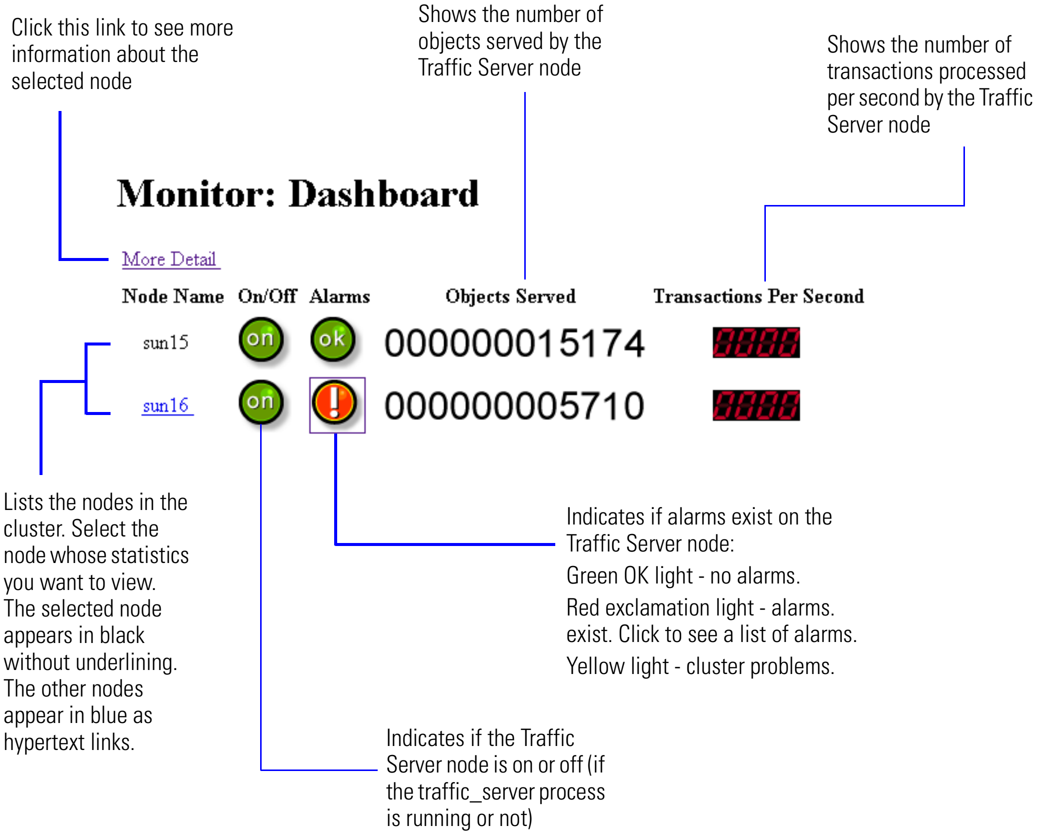
In Monitor mode, Traffic Manager displays a series of buttons on the **Monitor** tab. Each button represents a group of statistics. Click on a button to view its statistics. Each button is described briefly below.

Note All the statistics displayed in Monitor mode are described in detail in *Appendix A, Traffic Manager Statistics*.

The Dashboard button

Click the **Dashboard** button to see a concise view of your Traffic Server system, displaying all cluster nodes by name and tracking essential statistics for each node. If you want to display detailed information about a particular node, you can click the node's name on the **Dashboard**, and then click on one of the other buttons on the **Monitor** tab.

Figure 13 shows the Dashboard.



Click this link to see more information about the selected node

Shows the number of objects served by the Traffic Server node

Shows the number of transactions processed per second by the Traffic Server node

Monitor: Dashboard

More Detail

Node Name	On/Off	Alarms	Objects Served	Transactions Per Second
sun15	on	ok	000000015174	8888
sun16	on	!	000000005710	8888

Lists the nodes in the cluster. Select the node whose statistics you want to view. The selected node appears in black without underlining. The other nodes appear in blue as hypertext links.

Indicates if alarms exist on the Traffic Server node:

Green OK light - no alarms.

Red exclamation light - alarms. exist. Click to see a list of alarms.

Yellow light - cluster problems.

Indicates if the Traffic Server node is on or off (if the traffic_server process is running or not)

Figure 13 The Dashboard

The Node button

Click the **Node** button to see the following information about the selected Traffic Server node:

Note If the node is part of a cluster, two sets of statistics are shown: information about the single node and information showing an average value for all the nodes in the cluster.

- ✔ If the node is active or inactive
- ✔ The date and time that the `traffic_server` process on the Traffic Server node was started
- ✔ If the node is part of a cluster
- ✔ Cache performance information, such as the document hit rate, the bandwidth savings, and what percentage of the cache is currently free
- ✔ The number of client and server connections currently open and the number of transfers currently in progress
- ✔ Network information, such as the client throughput in Mbits per second and the number of transactions being processed per second
- ✔ Name resolution information, such as the host database hit rate and the number of DNS lookups per second

The name of each statistic on the **Node** page appears as a link (the text is underlined). Click a link to display the statistical information in a graph. You can display a single graph showing multiple statistics from the **Graphs** page. Refer to *The Graphs button* below.

The Graphs button

Click the **Graphs** button to view the same statistics displayed on the **Node** page (cache performance, current connections and transfers, network, and name resolution) in graphical format. You can display multiple statistics in one graph.

To display a single graph, click the graph's name in the list. Each graph's name appears as a link (The graphs that display are the same graphs that display when you click a link for a statistic on the **Node** page).

To display multiple statistics in one graph, click the box next to the name of each graph you want to display, then click the **Graph Selected** button.

The Protocols button

Click the **Protocols** button to see the following information about a Traffic Server node:

- ✔ HTTP statistics that show information about HTTP transactions and speeds (such as, cache misses, cache hits, connection errors, aborted transactions), and client and server connection information
- ✔ FTP statistics that include the number of open connections, successful PASV and PORT connections, and unsuccessful PASV and PORT connections
- ✔ NNTP statistics that include the number of client and server connections, the number of article and group hits and misses, and information about posts, pulls, and feeds
- ✔ ICP statistics that include information about queries originating from the Traffic Server node and from ICP peers (parents and siblings)
- ✔ WCCP version 1.0 or 2.0 statistics that include information about the routers being used, the number of active nodes, the leader's IP address, and whether WCCP is currently enabled on the Traffic Server node

Note If you have installed a plug-in with Traffic Server (for example, Media-IXT), the **Protocols** button displays information about additional protocols. Refer to the documentation that comes with the plug-in you are using for more information.

The Cache button

Click the **Cache** button to view the following statistics about the Traffic Server's cache:

- ✔ How much space in the cache is currently being used and the maximum cache size in GB
- ✔ The total size of the RAM cache in bytes, and the number of RAM cache hits and misses
- ✔ The number of cache lookups, object reads, writes, updates, and removes

The Other button

Click the **Other** button to view the following statistics about a Traffic Server node:

- ✔ The total number of lookups and hits in the host database, and the average lookup time
- ✔ The total number of lookups in the Domain Name Server (DNS), the number of successful lookups, and the average lookup time
- ✔ The number of nodes in the cluster, the total number of cluster operations, the number of bytes read and written to all the nodes in the cluster, and the current number of open connections in the cluster
- ✔ The number of successful and unsuccessful connections to the SOCKS server, and the number of connections currently in progress
- ✔ The number of log files currently open, the amount of spaced currently being used for log files, the number of access events and error events logged, and the number of access events skipped

The MRTG button

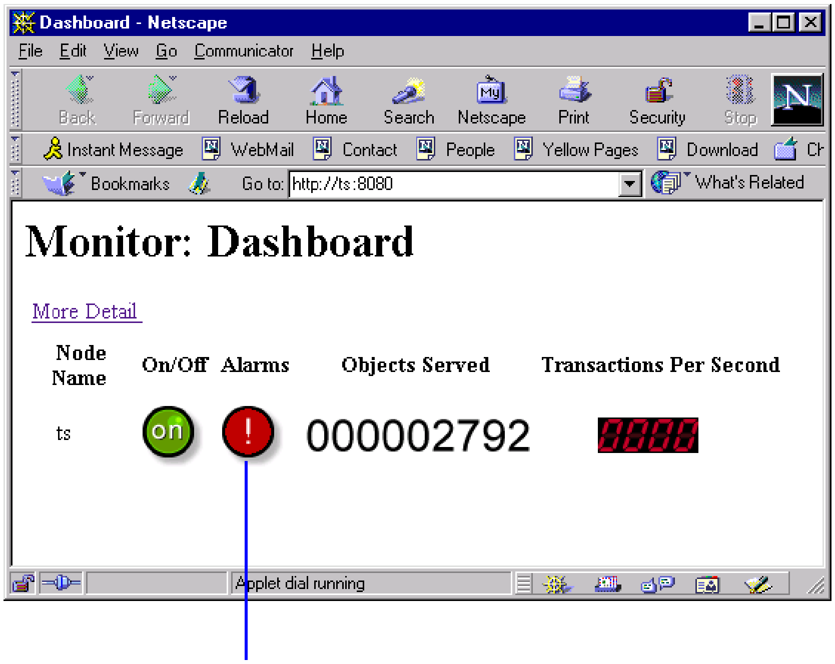
Displays MRTG graphs. Refer to *Using MRTG, on page 119*.

Working with Traffic Manager Alarms

Traffic Server signals an alarm when it detects a problem (for example, if the `traffic_server` process shuts down, if the space allocated to event logs is full, or if Traffic Server cannot write to a configuration file).

Traffic Server signals alarms by displaying a red alarm button with an exclamation point on the **Dashboard** in Traffic Manager. Click the red alarm button to view alarm messages.

Figure 14 shows the red alarm button on the Dashboard.


Dashboard - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Instant Message WebMail Contact People Yellow Pages Download Ch
Bookmarks Go to: http://ts:8080 What's Related
Applet dial running

Monitor: Dashboard

More Detail

Node Name	On/Off	Alarms	Objects Served	Transactions Per Second
ts	on	!	000002792	0000

This button indicates that one or more alarms exist on the Traffic Server node. Click this button to display alarm messages.

Figure 14 Alarms on the Dashboard

Resolving alarms

After you have read an alarm message, you can click the **Resolve** button in the alarm message window to tell Traffic Server that you have been informed of the problem and to dismiss the alarm. *Traffic Server alarm messages, on page 310* provides a description of all the alarm messages that Traffic Server provides.

Important Clicking the **Resolve** button only dismisses alarm messages; it does not actually resolve the cause of the alarms.

Configuring Traffic Server to E-mail alarms

Alarm messages are built into Traffic Server, you cannot change them. However, you can write a script file to execute certain actions when an alarm is signaled. For example, if Traffic Server signals an alarm to indicate that the logging directory is full, you can write a script file that sends an E-mail to alert someone of the problem.

The Traffic Server CD provides a sample script file named `example_alarm_bin.sh` (UNIX) or `example_alarm_bin.bat` (Windows). You can modify the file to suit your needs.

Viewing Statistics from Traffic Line

As an alternative to using Traffic Manager, you can use the Traffic Line command-line interface to view statistics about Traffic Server performance and web traffic. Traffic Line provides a quick way of viewing Traffic Server statistics if you do not have a browser installed on your machine. You can examine:

- ✔ Groups of statistics about a node or a cluster (for example, all statistics related to the cache)
- ✔ A single statistic about a node or a cluster (for example, the number of objects served from the cache)

In addition to viewing statistics, you can also configure a Traffic Server from Traffic Line and use batch mode commands to stop and restart a Traffic Server system. Refer to *Configuring Traffic Server using Traffic Line, on page 128* and *Appendix C, Traffic Line Commands*.

Viewing groups of statistics

Use Monitor mode in a Traffic Line Interactive session to view groups of statistics (for example, HTTP statistics that include information about HTTP transactions and speeds).

The statistics displayed in Traffic Line Monitor mode are the same as those displayed in Traffic Manager Monitor mode.

Starting Monitor mode

You access Monitor mode from a Traffic Line Interactive session.

▼ To start Monitor mode:

1. In UNIX, log in to a Traffic Server node as the Traffic Server administrator, then make Traffic Server's `bin` directory your working directory.

   In Windows, open a Command Prompt window, then `cd` to the `bin` directory (located in the Traffic Server installation directory).

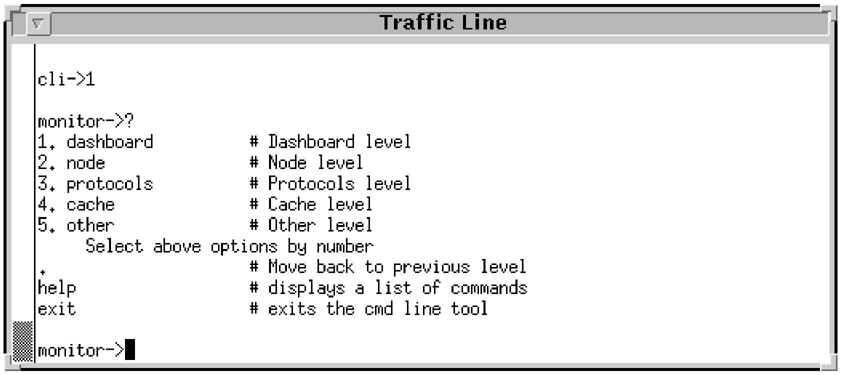
2. Enter the following command, and then press Return:

```
traffic_line -i
```

   The `cli->` prompt indicates that you are now in a Traffic Line Interactive session.

3. At the prompt, enter `1` to reach Monitor mode.

   The Monitor mode command list appears (shown below).



```
cli->1

monitor->?
1. dashboard          # Dashboard level
2. node               # Node level
3. protocols          # Protocols level
4. cache              # Cache level
5. other              # Other level
     Select above options by number
.                     # Move back to previous level
help                  # displays a list of commands
exit                  # exits the cmd line tool

monitor->
```

Note If the command list does not display, enter `?` at the prompt.

4 At the prompt, enter the number that corresponds to the group of statistics you want to view. For example, to view protocol related statistics, enter `3`, then press Return.

   For a description of the type of information listed in each group, refer to *page 112*.

Navigating Monitor mode

Monitor mode in a Traffic Line Interactive session consists of several levels of commands. Each command has a number associated with it. To start a command, enter its number at the command prompt and press Return.

You can enter `?` (a question mark) at the prompt and press Return at any time to display the list of commands available at the current level.

To return to a previous command level, enter `.` (a period) at the prompt and press Return.

To exit a Traffic Line Interactive session, enter `exit` at the prompt and press Return.

Retrieving individual statistics

You can view specific information about a Traffic Server node or cluster by specifying the variable that corresponds to the statistic you want to see. Using this method, you see only the information you want to obtain instead of seeing a group of related statistics.

▼ To retrieve a single statistic:

1 In UNIX, log in to a Traffic Server node as the Traffic Server administrator, then make Traffic Server's `bin` directory your working directory.

   In Windows, open a Command Prompt window, then `cd` to the `bin` directory (located in the Traffic Server installation directory).

2 Execute the following command, then press Return:

```
traffic_line -i
```

   The `cli->` prompt indicates that you are now in a Traffic Line Interactive session.

3 Enter the following command:

```
get variable
```

   where *`variable`* is the variable that represents the information you want to retrieve. For a list of the variables you can specify, refer to *Appendix C, Traffic Line Commands*.

   For example, the following command displays the document hit rate for the Traffic Server node:

```
get proxy.node.http.cache_hit_ratio
```

Using MRTG

MRTG (Multi Router Traffic Grapher) is a graphing tool that enables you to monitor Traffic Server's performance and analyze network traffic. MRTG provides a variety of graphs that show information about virtual memory usage, client connections, document hit rates, hit and miss rates, and so on. MRTG uses five minute intervals to formulate the statistics and provides useful historical information.

You access MRTG from the **Monitor** tab in the Traffic Manager UI.

Accessing MRTG

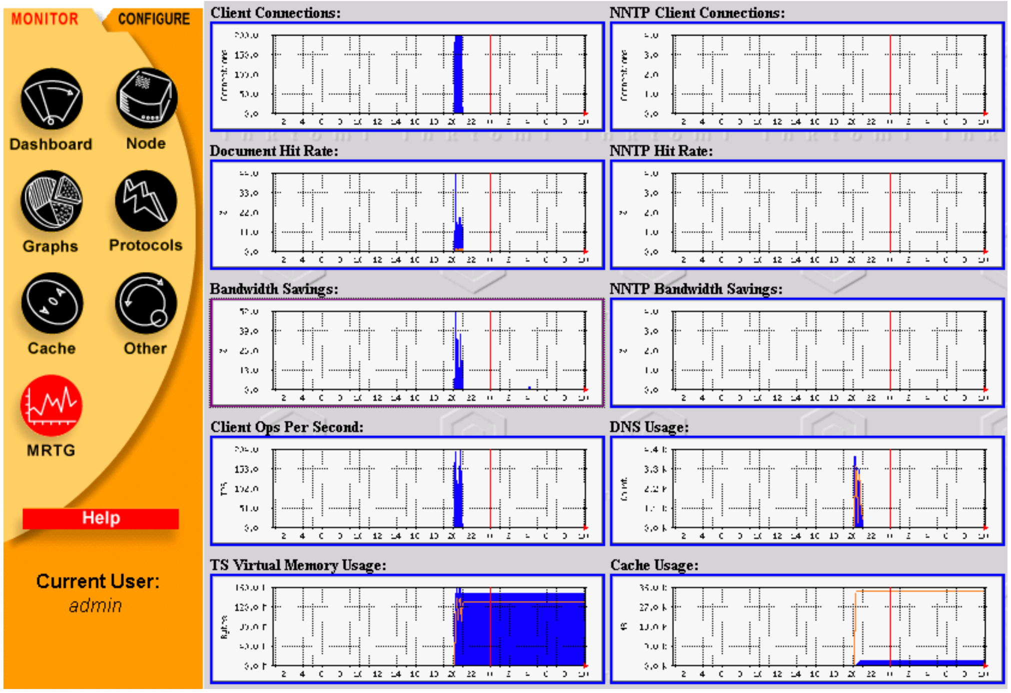
▼ To access MRTG:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 If your Traffic Server node is in a cluster, choose the Traffic Server node whose statistics you want to view from the **Dashboard** on the **Monitor** tab.

3 On the **Monitor** tab, click the **MRTG** button.

The MRTG index page opens. The figure below shows the MRTG Index page.



Note If MRTG has not been configured, the message `MRTG is not available` displays on the **MRTG** page. Follow the steps in *Configuring MRTG* below.

Configuring MRTG

If the message `MRTG is not available` displays on the MRTG page when you click the **MRTG** button on the **CONFIGURE** tab, MRTG has not been configured. Follow the steps bellow to configure the MRTG graphing tool.

▼ To configure MRTG (UNIX only):

1. Make sure `perl` is installed on your system.
2. Change directories to Traffic Server's `bin` directory.
3. The `perl` binary needs to be in your PATH. Type the following at the command prompt:

   ```
   perl ./pathfix.pl 'which perl'
   ```

4. Modify the MRTG update interval by typing the following at the command prompt:

   ```
   ./update_mrtg;sleep 5;./update_mrtg;sleep 5;
   ```

   By default, an MRTG update interval is set to 15 minutes. This command sets the update to 5 minutes.
5. Start the `mrtg cron` updates by typing the following command:

   ```
   ./mrtgcron start
   ```

6. Wait about 15 minutes before accessing MRTG from the Traffic Manager UI.

Note To stop `mrtg cron` updates, type the command `./mrtgcron stop`.

Navigating MRTG

The MRTG index page shows a subset of the graphs available for display. Click on a graph to see daily, weekly, monthly, and yearly statistics for that particular graph.

Click on the **more info** button to view a detailed description of the graphs.

You can also click on the **daily view** link at the bottom of the index page to see daily Traffic Server statistics and on the **weekly overview** link to see weekly Traffic Server statistics. Clicking on these links provides a more extensive selection of related graphs.

Creating your own MRTG index page

You can create your own MRTG index page displaying only the graphs you want to view.

Note The following procedure is for Netscape Navigator only.

▼ To create your own index page:

1. From the browser's **File** menu, select **New/Blank Page**.
2. From the MRTG Index page, or from the daily view or weekly view page, click a graph and drag it to the blank page.
3. To change the size of a graph on your index page, select the graph and then select **Image Properties** from the **Format** menu. Change the height and width in the **Dimensions** area.
4. Save the Index page as `index.html` in Traffic Server's `ui/mrtg` directory.

Using SNMP

The Simple Network Management Protocol (SNMP) is a standard protocol used for network management. SNMP agents collect and store management information in Management Information Bases (MIBs), and SNMP managers can probe the agents for this information. In addition, SNMP agents can send alarms and alerts called *SNMP traps* to the SNMP manager to warn of any problems.

Traffic Server's SNMP agent supports access to two management information bases (MIBs): MIB-2 (a standard MIB) and the Inktomi Traffic Server MIB. Descriptions of the Traffic Server MIB variables are provided in the `inktomi-ts-mib.my` file in Traffic Server's `config/mibs` directory. The Traffic Server MIB contains both node-specific and cluster-wide information.

To use SNMP on your Traffic Server system, you need to:

✔ Enable Traffic Server's SNMP agent (see *Enabling SNMP* below)

✔ Configure Traffic Server to send SNMP traps

✔ Control MIB access to specific hosts

Note For the Traffic Server SNMP agent to respond to requests from SNMP managers and send SNMP traps, the SNMP daemon must be running on your Traffic Server system.

Enabling SNMP

Traffic Server's SNMP agent must be enabled so that SNMP managers can access the MIBs and gather information.

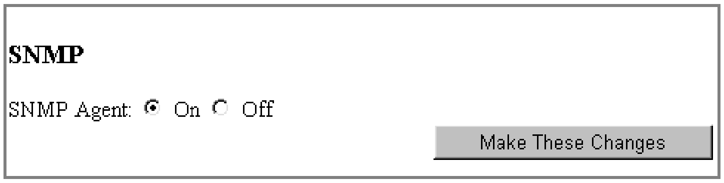
You can enable the SNMP agent by using the Traffic Manager UI or by editing a configuration file manually. Both procedures are provided below.

▼ To enable the SNMP agent from Traffic Manager:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Server** button.

3 Scroll to the **SNMP** section of the **Server Basics** page (shown below).



SNMP

SNMP Agent: ● On ○ Off

4 Click the **SNMP Agent On** button.

5 Click the **Make These Changes** button.

▼ To enable the SNMP agent manually:

1 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

2 Edit the following variable:

Variable	Description
proxy.config.snmp.master_agent_enabled	Set this variable to 1 to enable SNMP on the Traffic Server node.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

Configuring SNMP trap destinations

To configure SNMP trap destinations, edit the `snmpd.cnf` file located in Traffic Server's `config` directory. Refer to *snmpd.cnf, on page 291*.

Controlling MIB access

By default, read-only access to the Traffic Server MIBs is granted to any host that makes SNMP requests using the community string `public`. Inktomi recommends that you configure your Traffic Server system to control MIB access so that only certain hosts can access SNMP information.

To configure Traffic Server to control MIB access, edit the `snmpd.cnf` file located in Traffic Server's `config` directory. Refer to *snmpd.cnf, on page 291*.

Chapter 10

Configuring Traffic Server

Traffic Server provides several options for configuring the system.

This chapter discusses the following topics:

- *Configuring Traffic Server using the Traffic Manager UI, on page 124*
- *Configuring Traffic Server using Traffic Line, on page 128*
- *Configuring Traffic Server using configuration files, on page 130*

Configuring Traffic Server using the Traffic Manager UI

You can use the Traffic Manager UI to view and change your Traffic Server configuration. You set configuration options using Traffic Manager's Configure mode.

Note Certain Traffic Server configuration options can only be changed by editing configuration variables either in the `records.config` file or from the Traffic Line command-line interface. Refer to *Setting configuration options in batch mode, on page 129* and *Configuring Traffic Server using configuration files, on page 130*.

Starting Traffic Manager Configure mode

▼ To start Traffic Manager Configure mode:

1 Open your web browser.

The Traffic Manager requires Java and JavaScript; be sure to enable Java and JavaScript in your browser.

2 Type one of the following locations in your browser:

Standard

```
http://nodename:adminport/
```

SSL

```
https://nodename:adminport/
```

where `nodename` is the name of the Traffic Server node and `adminport` is the number assigned to the Traffic Manager port.

Note Use the SSL `https` command to access Traffic Manager only if you have restricted access to Traffic Manager via SSL connections; otherwise, use the standard `http` command.

3 If necessary, log on to Traffic Server with the administrator ID and password, or your administrator account.

Note The administrator ID and password are set during Traffic Server installation. You can change the ID and password, as well as create and modify administrator accounts. For more information, refer to *Controlling access to the Traffic Manager UI, on page 136*.

The Traffic Manager UI starts by default in Monitor mode.

4 Click the **Configure** tab to display the Configure mode buttons (shown below).

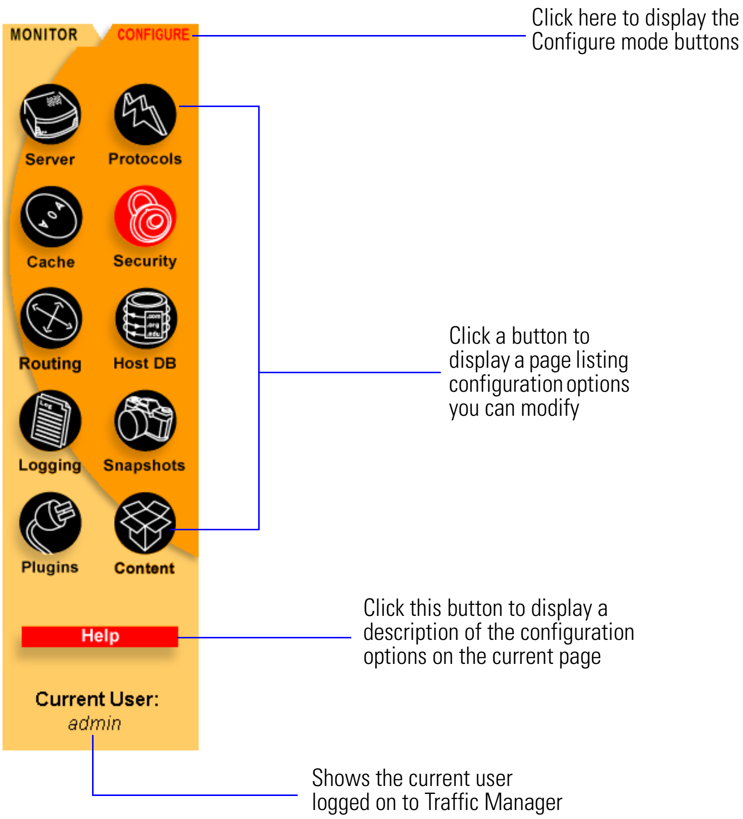

MONITOR
CONFIGURE
Server
Protocols
Cache
Security
Routing
Host DB
Logging
Snapshots
Plugins
Content
Help
Current User:
admin
Click here to display the Configure mode buttons
Click a button to display a page listing configuration options you can modify
Click this button to display a description of the configuration options on the current page
Shows the current user logged on to Traffic Manager

Figure 15 Traffic Manager Configure mode buttons

Using Configure mode

In Configure mode, Traffic Manager displays a series of buttons on the **Configure** tab. Each button represents a group of configuration options. Each button is described below.

Note All the configuration options available in Configure mode are described in *Appendix B, Traffic Manager Configuration Options*.

The Server button

Click the **Server** button to view or change Traffic Server's basic configuration options. You can:

- ✔ Turn the Traffic Server on or off
- ✔ Identify the hostname of the Traffic Server, the Traffic Server port and user ID
- ✔ Enable local domain expansion and .com domain expansion
- ✔ Restart the Traffic Manager process, change the Traffic Manager port, and edit the refresh rate for the statistics displayed in Monitor mode
- ✔ Configure the use of virtual IP addresses
- ✔ Auto-configure browsers to connect to Traffic Server as a proxy server
- ✔ Restrict the number of network connections Traffic Server will accept (throttling of network connections)

- ✔ Configure the way Traffic Server handles overload conditions in transparency mode (load shedding)
- ✔ Enable SNMP
- ✔ Configure customizable response pages for HTTP transactions

The Protocols button

Click the Protocols button to view or change Traffic Server's protocol configuration. You can:

- ✔ Tune HTTP time-outs and remove HTTP headers to maintain the privacy of your site and users
- ✔ Configure how Traffic Server caches and serves news articles (NNTP)
- ✔ Configure Traffic Server to restrict SSL connections to certain ports
- ✔ Set FTP options, such as the connection mode, inactivity timeouts, and the anonymous FTP password

The Cache button

Click the **Cache** button to view or change Traffic Server's cache configuration. You can:

- ✔ Enable/disable HTTP, NTTP, and FTP caching
- ✔ Configure Traffic Server to ignore user requests to bypass the cache
- ✔ Set cache storage options, such as the maximum HTTP/FTP object size, the maximum number of alternates that Traffic Server is allowed to cache, and view a list of the files or disk partitions allotted to cache storage and their sizes
- ✔ Configure HTTP and FTP object freshness options
- ✔ Configure variable content options

The Security button

Click the **Security** button to view or change Traffic Server's security options. You can:

- ✔ Configure access to the Traffic Manager UI by setting an administrator ID and password, and creating administrator accounts
- ✔ Configure Traffic Server integration into your firewall and control traffic through the SOCKS server

The Routing button

Click the **Routing** button to view or change Traffic Server routing options. You can:

- ✔ Enable HTTP parent caching and identify the HTTP parent cache(s) you want to use
- ✔ Configure Traffic Server to be part of an ICP cache hierarchy
- ✔ Enable reverse proxy and set mapping rules
- ✔ Check if the Traffic Server is running in transparent proxy mode
- ✔ Check if WCCP is enabled

The Host DB button

Click the **Host Database** button to view or edit Traffic Server's host database and DNS configuration. You can:

- ✔ Set host database timeouts
- ✔ Set how long Traffic Server must wait for the DNS server to respond to a request and how many times Traffic Server must retry a DNS lookup.

The Logging button

Click the **Logging** page to view or change Traffic Server logging options. You can:

- ✔ Enable/disable event logging
- ✔ Control where log files are located, how much disk space they can consume, and how low disk space in the logging directory is handled
- ✔ Choose a central location for storing and collating log information
- ✔ Choose standard log file formats
- ✔ Set log splitting options
- ✔ Enable custom logging and choose the custom log format
- ✔ Configure when and how to roll log files

The Snapshots button

Click the **Snapshots** button to take a snapshot of the current configuration values or restore previously saved configuration values. One configuration snapshot consists of a complete set of Traffic Server configuration files.

The Plugins button

Click the **Plugins** button to list the plugins currently running on your Traffic Server that are configurable from the Traffic Manager UI. A plugin is a program that extends the functionality of Traffic Server. For example, plugins can perform web server blacklisting, web content filtering, user authentication, and data transformation.

The Content button

Click the **Content** button to view or change the list of objects that Traffic Server is scheduled to update automatically in the local cache. You can instruct Traffic Server to explicitly preload objects in to the cache, thereby increasing Traffic Server performance.

Configuring Traffic Server using Traffic Line

You can use Traffic Line to view and change your Traffic Server configuration as an alternative to using the Traffic Manager UI. The advantage of using Traffic Line is that it is a command-line interface, therefore, you do not need to have a browser installed on the remote system from which you want to perform the configuration.

You can set configuration options using Traffic Line's Configure mode or Traffic Line's batch mode.

Starting Configure mode

Configure mode is accessed from a Traffic Line Interactive session.

▼ To start Configure mode:

1 In UNIX, log in to a Traffic Server node as the Traffic Server administrator, then make Traffic Server's `bin` directory your working directory.

In Windows, open a Command Prompt window, then `cd` to the `bin` directory (located in the Traffic Server installation directory).

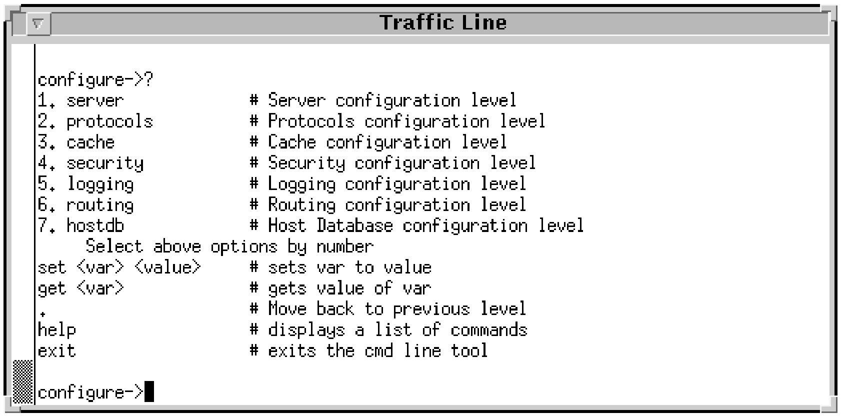
2 Enter the following command, and then press Return:

```
traffic_line -i
```

The `cli->` prompt indicates that you are now in a Traffic Line Interactive session.

3 At the prompt, enter `2` to reach Configure mode.

The Configure mode command list appears (shown below).



```
configure->?
1. server            # Server configuration level
2. protocols         # Protocols configuration level
3. cache             # Cache configuration level
4. security          # Security configuration level
5. logging           # Logging configuration level
6. routing           # Routing configuration level
7. hostdb            # Host Database configuration level
     Select above options by number
set <var> <value>    # sets var to value
get <var>            # gets value of var
.                    # Move back to previous level
help                 # displays a list of commands
exit                 # exits the cmd line tool

configure->
```

If the command list does not display, enter ? at the prompt.

Navigating Configure mode

Configure mode in a Traffic Line interactive session consists of levels of commands. Each command has a number associated with it. To execute a command, enter its number at the command prompt, then press Return.

You can enter `?` (a question mark) at the prompt and press Return at any time to display a list of commands at the current level.

To return to a previous command level, enter `.` (a period) at the prompt and press Return.

To exit a Traffic Line interactive session, enter `exit` at the prompt and press Return.

Setting configuration options in Configure mode

▼ To set configuration options in Configure mode:

1. Start Configure mode as described in *Starting Configure mode, on page 128*.
2. At the prompt, enter the number that corresponds to the type of configuration you want to change, then press Return. For example, to change protocol configuration, enter 2 at the prompt.

   Each command displays a sub level of commands that group specific configuration variables together. For example, the `Protocols` command (2) displays the following sub level of commands: `display`, `http`, `ftp`, and `nntp`.
3. To view a group of configuration options enter the number of the sub level command. Each configuration option has a number associated with it.
4. To change a configuration option, enter the following command:

   ```
   change no value
   ```

   where *`no`* is the number associated with the configuration option and *`value`* is the value you want to set.

   For example, to change the FTP inactivity timeout option to 200 seconds, go the `Protocol/ftp` command level, then enter `change 15 200` at the prompt. 15 is the number associated with the FTP inactivity timeout configuration option.

For a detailed description of all the configuration options available in Traffic Line, refer to *records.config, on page 258*.

Setting configuration options in batch mode

You can also set configuration options from Traffic Line batch mode.

▼ To set configuration options in batch mode:

1. In UNIX, log in to a Traffic Server node as the Traffic Server administrator, then make Traffic Server's `bin` directory your working directory.

   In Windows, open a Command Prompt window, then `cd` to the `bin` directory (located in the Traffic Server installation directory).
2. Enter the following command:

   ```
   traffic_line -s var -v value
   ```

   where *`var`* is the variable associated with the configuration option (for a list of the variables, refer to *Appendix C, Traffic Line Commands*) and *`value`* is the value you want to use.

   For example, to change the FTP inactivity timeout option to 200 seconds, enter the following command at the prompt and press Return:

   ```
   traffic_line -s proxy.config.ftp.control_connection_timeout -v 200
   ```

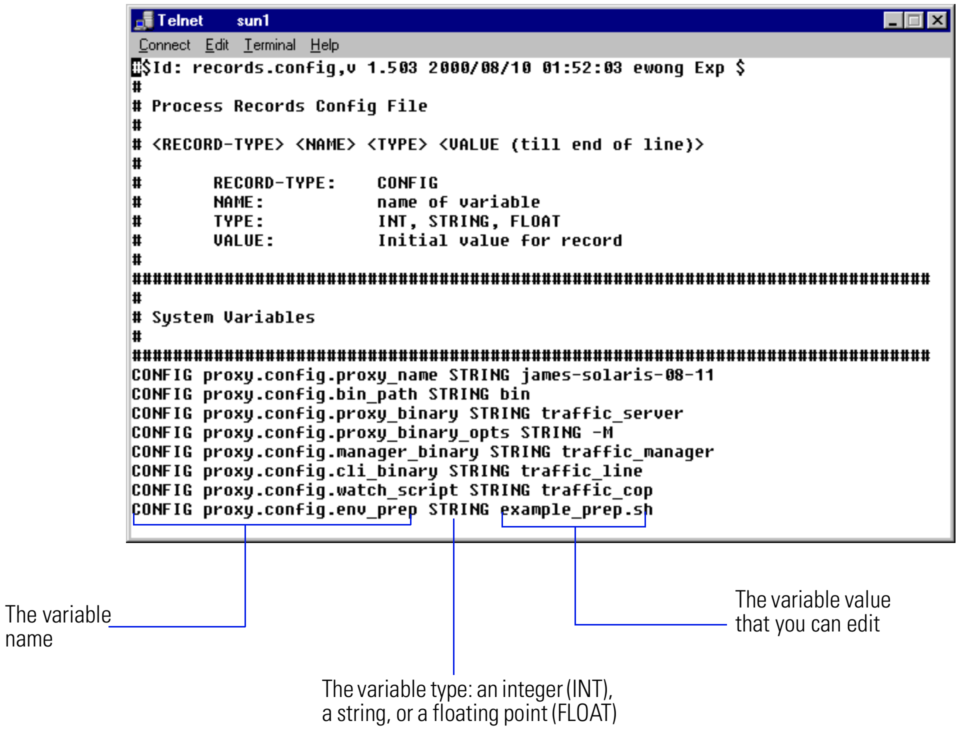
Configuring Traffic Server using configuration files

As an alternative to using Traffic Manager or Traffic Line, you can change Traffic Server configuration options by manually editing specific variables in the `records.config` file.

The `records.config` file is located in Traffic Server's `config` directory. To edit the variables, open the file in a text editor (such as `vi` or `emacs`) and change the variable value.

Note After you modify the `records.config` file, Traffic Server has to reread the configuration files. From Traffic Server's `bin` directory, enter the Traffic Line batch mode command `traffic_line -x`. In some cases, you have to restart Traffic Server for the changes to take effect.

The following is a sample portion of the `records.config` file.


Telnet sun1
Connect Edit Terminal Help
The variable name
The variable type: an integer (INT), a string, or a floating point (FLOAT)
The variable value that you can edit

```
$Id: records.config,v 1.503 2000/08/10 01:52:03 ewong Exp $
#
# Process Records Config File
#
# <RECORD-TYPE> <NAME> <TYPE> <VALUE (till end of line)>
#
#       RECORD-TYPE:    CONFIG
#       NAME:           name of variable
#       TYPE:           INT, STRING, FLOAT
#       VALUE:          Initial value for record
#
##############################################################################
#
# System Variables
#
##############################################################################
CONFIG proxy.config.proxy_name STRING james-solaris-08-11
CONFIG proxy.config.bin_path STRING bin
CONFIG proxy.config.proxy_binary STRING traffic_server
CONFIG proxy.config.proxy_binary_opts STRING -M
CONFIG proxy.config.manager_binary STRING traffic_manager
CONFIG proxy.config.cli_binary STRING traffic_line
CONFIG proxy.config.watch_script STRING traffic_cop
CONFIG proxy.config.env_prep STRING example_prep.sh
```

Figure 16 A sample records.config file

In addition to the `records.config` file, Traffic Server provides other configuration files that are used to configure specific features. All the configuration files are described in *Appendix D, Configuration Files*.

Chapter 11

Security Options

Traffic Server provides a number of security features.

This chapter discusses the following topics:

Traffic Server security options

Traffic Server provides numerous options that enable you to establish secure communication between the Traffic Server system and other computers on the network. Using the security options, you can:

- ✔ Control which clients are allowed to access the Traffic Server proxy cache. Refer to *Controlling client access to the Traffic Server proxy cache, on page 133*.
- ✔ Control which hosts are allowed to access the Traffic Server machine. Refer to *Controlling host access to the Traffic Server machine (ARM security), on page 134*.
- ✔ Control and secure access to the Traffic Manager UI using:
    - ✗ Administrator accounts (refer to *Setting the administrator ID and password, on page 136* and *Creating a list of administrator accounts, on page 138*)
    - ✗ An access control list that defines which hosts are allowed to access the Traffic Manager (refer to *Controlling host access to the Traffic Manager UI, on page 139*)
    - ✗ SSL (Secure Sockets Layer) protection for encrypted, authenticated access (refer to *Using SSL for secure administration, on page 140*)
- ✔ Configure Traffic Server integration into your firewall and control traffic through the SOCKS server. Refer to *Configuring SOCKS firewall integration, on page 142*.
- ✔ Configure Traffic Server to use multiple DNS servers to match your site's security configuration. Refer to *Configuring DNS server selection (split DNS), on page 145*.
- ✔ Configure Traffic Server to use LDAP-based proxy authentication. Refer to *Configuring LDAP-based proxy authentication, on page 146*.
- ✔ Secure reverse proxy connections between a client and Traffic Server and Traffic Server and an origin server, using the SSL termination option. Refer to *Using SSL Termination, on page 147*.

Controlling client access to the Traffic Server proxy cache

You can configure Traffic Server to allow only certain clients to use the proxy cache.

▼ To specify the clients allowed to use Traffic Server as a proxy cache:

1 In a text editor, open the `ip_allow.config` file located in Traffic Server's `config` directory.

By default, the file contains the following line that allows all clients to access the Traffic Server proxy cache:

```
src_ip=0.0.0.0-255.255.555.255    action=ip_allow
```

2 Comment out the default line, as shown below.

```
#src_ip=0.0.0.0-255.255.555.255    action=ip_allow
```

3 Using the following format, add a line for each IP address or range of IP addresses allowed to access Traffic Server:

```
src_ip=IP address or range of IP addressses    action=ip_allow
```

where *`IP address or range of IP addresses`* is the IP address or range of IP addresses of the clients that are allowed to access the Traffic Server proxy cache.

For example, to allow client access from a host with the IP address 11.11.11.1 and from a host with the IP address 1.1.1.1, enter the following lines in the file:

```
src_ip=11.11.11.1    action=ip_allow
src_ip=1.1.1.1    action=ip_allow
```

4 Save and close the `ip_allow.config` file.

5 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

6 Run the command `traffic_line -x` to apply the configuration changes.

Note If an unauthorized client tries to access Traffic Server, a message displays in their browser indicating that the requested content cannot be obtained. For example, in Netscape Version 4.7, the message *The document contained no data* appears in the browser window. In Internet Explorer Version 5.0, the message *The page cannot be displayed* appears in the browser window.

Controlling host access to the Traffic Server machine (ARM security)

For security reasons, you might want to restrict the type of communication possible with machines running Traffic Server. Using Traffic Server's ARM security option, you can create an access control list that is used to either allow or deny other hosts from communicating with the Traffic Server machine on specific ports. This *firewall* prevents potentially malicious packets from disrupting the operation of the machine.

When the ARM security option is enabled, the Traffic Server ARM examines UDP and TCP packets as they arrive at the Traffic Server machine and matches them against the access control list that you specify in a configuration file. The ARM checks all UDP packets (since UDP communication is, by definition, connectionless) and looks at the first TCP packet initiating the session against the configuration file access control list. Acceptable packets using either protocol are then passed up the network stack. Only incoming UDP and TCP packets are affected. This means that it is always possible to initiate TCP and UDP connections from the Traffic Server regardless of the access control list configured.

To use the ARM security feature, you must do the following in the order listed:

✔ Edit the `arm_security.config` file to open specific ports and define the hosts that are allowed to communicate with the Traffic Server machine.

Important By default, the `arm_security.config` file specifies that all ports on the Traffic Server machine are closed (including telnet) except port 8080, which remains open to allow Traffic Server to continue functioning normally. If you enable the ARM security option with the default `arm_security.config` file, you will be locked out of the system. Before you enable the ARM security option, ensure that you have either console access to the Traffic Server machine, or that you have added the appropriate rules to the `arm_security.config` file to allow `telnet` or `ssh` access for yourself.

✔ Enable the ARM security option

▼ To edit the arm_security.config file and enable the ARM security option:

1 In a text editor, open the `arm_security.config` file located in Traffic Server's `config` directory.

2 Add open, allow, and deny rules to define which ports you want to remain open and which hosts are allowed to communicate with Traffic Server.

Each rule must have one of the following formats:

```
open tcp|udp ports o_ports
```

```
deny tcp|udp dport d_ports src src_IP_addresses
```

```
allow tcp|udp dport d_ports src src_IP_addresses
```

`where` *`o_ports`* is the port, or series of ports separated by spaces, that you want to remain open.

`d_ports` is the destination port, or series of destination ports separated by spaces, through which TCP or UDP traffic should either be allowed or denied.

`src_IP_addresses` is the IP address or range of IP addresses specifying the source of the communication.

Note If the Traffic Server machine is part of a cluster, ensure that port 90 is open for UDP traffic and include rules to allow communication from all other machines in the cluster.

You may also want to open the NFS and DNS ports, if required.

The following example rules specify that ports 119, 23, and 554 are to remain open for TCP communication and that hosts 1.1.1.1 through 1.1.1.7 are allowed access to destination port 80. However, the host 11.11.11.11 is denied access to destination port 80.

```
open tcp ports 119 23 554
allow tcp dport 80 src 1.1.1.1-1.1.1.7
deny tcp dport 80 11.11.11.11
```

For more information about the format of the `arm_security.config` file and additional options that can be used, refer to *arm_security.config, on page 228*.

3 Save and close the `arm_security.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

6 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

7 Edit the following variable:

Variable	Description
proxy.config.arm.security_enabled	Set this variable to 1 to enable ARM security.

Note To disable the ARM security option, set the proxy.config.arm.security_enabled variable to 0 (zero).

8 Save and close the `records.config` file.

9 Restart Traffic Server.

Controlling access to the Traffic Manager UI

You can restrict access to the Traffic Manager UI to ensure that only authenticated users can change Traffic Server configuration options and view performance and network traffic statistics. You can:

- ✔ Set an administrator ID and password. A user that logs in to Traffic Manager with the administrator ID has access to all Traffic Manager activities. (See *Setting the administrator ID and password* below.)
- ✔ Create and maintain a list of administrator accounts that determines who can log into the Traffic Manager and which activities they can perform. (See *Creating a list of administrator accounts, on page 138*)
- ✔ Create an access control list of IP addresses that defines which machines can access the Traffic Manager UI. (See *Controlling host access to the Traffic Manager UI, on page 139*)
- ✔ Use SSL for secure administration (see *Using SSL for secure administration, on page 140*).

Setting the administrator ID and password

During Traffic Server installation, you assign an administrator ID and password that controls access to the Traffic Manager UI. A user that logs on to Traffic Manager using the correct ID and password can view all the statistics on the **Monitor** tab and change any configuration options on the **Configure** tab.

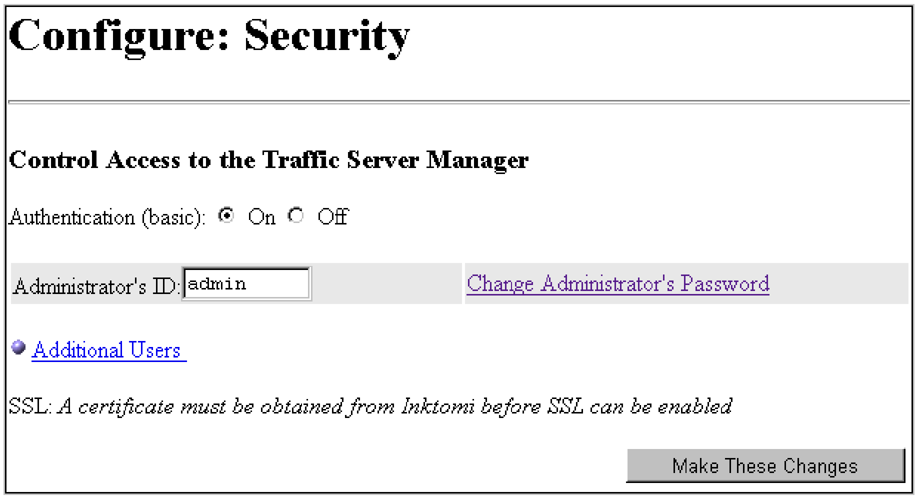
You can change the administrator ID and password at any time.

▼ To change the administrator ID and password:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Security** button.

The Security page opens displaying the **Control Access to the Traffic Server Manager** section (shown below).



Configure: Security

Control Access to the Traffic Server Manager

Authentication (basic): ● On ○ Off

Administrator's ID: admin

Change Administrator's Password

Additional Users

SSL: *A certificate must be obtained from Inktomi before SSL can be enabled*

3 Select **Authentication (basic): On** to check the administrator ID and password when a user tries to access Traffic Manager from a browser.

When **Authentication (basic)** is **Off**, any user can access the Traffic Manager unless you have set up a list of IP addresses that are denied access to the Traffic Manager UI (refer to *Controlling host access to the Traffic Manager UI, on page 139*).

4 To change the current administrator ID, type a new ID in the **Administrator's ID** field.

5 To change the current password, click the **Change Administrator's Password** link, and then enter the current and new password in the boxes provided.

If you have forgotten the current administrator password, refer to *If you forget the administrator password* below.

6 Click the **Make These Changes** button to apply the configuration changes.

If you forget the administrator password

During installation, you can specify an administrator password. The installer automatically encrypts the password and stores the encryptions in the `records.config` file. Each time you change passwords in the Traffic Manager UI, Traffic Server updates the `records.config` file.

If you forget the administrator password and cannot access the Traffic Manager UI, you can clear the current password in the `records.config` file (set the value of the configuration variable to `NULL`), and then enter a new password in Traffic Manager. You cannot set passwords in the `records.config` file because the password variables can only contain password encryptions or the value `NULL`.

▼ To clear and re-enter the administrator password:

1 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

2 Edit the following variables:

Variable	Description
`proxy.config.admin.basic_auth`	Set this variable to 1 to enable authentication.
`proxy.config.admin.admin_password`	Change the value of this variable to `NULL` to leave the password blank.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

6 Log on to the Traffic Manager UI. When prompted for the user name and password, enter the administrator ID and leave the password entry blank.

Because you have already cleared the password in the `records.config` file, you do not need a password to log on as the administrator.

7 On the Traffic Manager **Configure** tab, click the **Security** button.

8 In the **Control Access to the Traffic Server Manager** section, click the **Change administrator's password** link.

9 Leave the **Old Password** field empty. Enter the new password in the **New Password** field, then re-enter the new password in the **New Password (again)** field.

10 Click the **Make This change** button.

Creating a list of administrator accounts

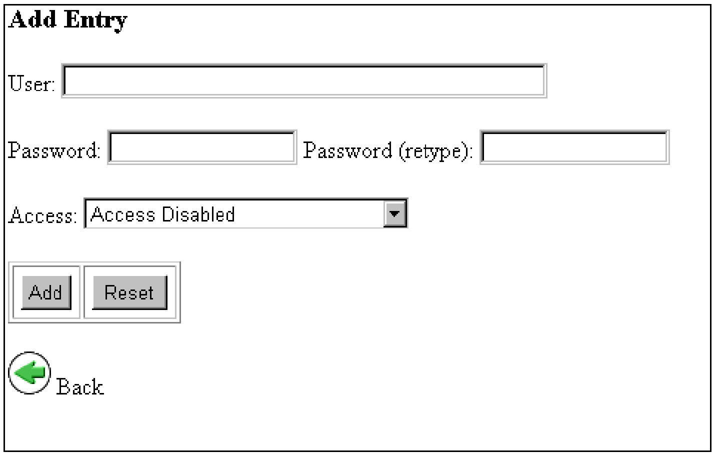
If a single administrator ID and password for the Traffic Manager UI is not sufficient security for your needs, you can create a list of administrator accounts that define who has access to the Traffic Manager and which activities they can perform.

You can use administrator accounts in addition to using the administrator ID and password.

▼ To create a list of administrator accounts:

1. From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).
2. On the **Configure** tab, click the **Security** button.

   The Security page opens displaying the **Control Access to the Traffic Server Manager** section at the top of the page.
3. Click the **Additional Users** link to open the Additional Users page.
4. Click the **Add Entry** button to open the Add Entry dialog box (shown below).



Access: Access Disabled

5. In the **User** field, enter the name of the user allowed to access Traffic Manager.
6. In the **Password** field, enter the password for the user, then enter the password again in the **Password (retype)** field.
7. In the **Access** drop-down list, select which Traffic Manager activities the user can perform:

   ✗ Select **Access Disabled** to disable Traffic Manager access for the user.

   ✗ Select **Monitor Only** to allow the user to view statistics from the **Monitor** tab only.

   ✗ Select **Monitor and View Configuration** to allow the user to view statistics from the **Monitor** tab and to *view* configuration options from the **Configure** tab.

   ✗ Select **Monitor and Modify Configuration** to allow the user to view statistics from the **Monitor** tab and to *change* configuration options from the **Configure** tab.

8. Click the **Add** button.
9. Repeat *step 4* through *step 8* for each user allowed to access Traffic Manager.
10. Click the **Make These Changes** button.
11. Click the **Configure Security link** to return to the **Control Access to the Traffic Server Manager** section of the **Security** page.
12. Select **Authentication (basic): On** to enable authentication.

    Traffic Server checks user names and passwords only if this option is enabled.
13. Click the **Make These Changes** button.

Controlling host access to the Traffic Manager UI

In addition to using an administrator ID and accounts, you can control which hosts have access to the Traffic Manager UI.

▼ To control which hosts can access the Traffic Manager UI:

1. In a text editor, open the `mgmt_allow.config` file located in Traffic Server's `config` directory.

   By default, the file contains the following line that allows all hosts to access the Traffic Manager UI:

   ```
   src_ip=0.0.0.0-255.255.555.255    action=ip_allow
   ```

2. Comment out the default line, as shown below.

   ```
   #src_ip=0.0.0.0-255.255.555.255    action=ip_allow
   ```

3. Using the following format, add a line for each IP address or range of IP addresses allowed to access the Traffic Manager UI:

   ```
   src_ip=IPaddress or range of IPaddressses   action=ip_allow
   ```

   where *`IPaddress or range of IPaddresses`* is the IP address or range of IP addresses of the hosts allowed to access the Traffic Manager UI.

   For example, to allow a host with the IP address 11.11.11.1 and a host with the IP address 1.1.1.1 to access the Traffic Manager UI, enter the following lines in the `mgmt_allow.config` file:

   ```
   src_ip=11.11.11.1    action=ip_allow
   src_ip=1.1.1.1    action=ip_allow
   ```

4. Save and close the `ip_allow.config` file.
5. In *UNIX*, make Traffic Server's `bin` directory your working directory.

   In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).
6. Run the command `traffic_line -x` to apply the configuration changes.

Using SSL for secure administration

Traffic Server supports the Secure Sockets Layer protocol (SSL) to provide protection for remote administrative monitoring and configuration using Traffic Manager. SSL security provides authentication of both ends of a network connection using certificates and provides privacy using encryption.

To use SSL, you must:

- ✔ Obtain an SSL certificate
- ✔ Enable the Traffic Manager SSL option
- ✔ Access Traffic Manager using the `https` command

Obtaining an SSL Certificate

You can obtain an SSL certificate from either:

- ✔ Inktomi Technical Support

  Use your websupport access account to obtain an SSL certificate. The certificate is a text file that you must install in Traffic Server's `config` directory. Each time you connect to Traffic Manager from your browser using the SSL certificate you obtained from Inktomi, you must go through an interactive acceptance dialogue.

- ✔ A recognized certificate authority (for example VeriSign)

  Install the certificate in Traffic Server's `config` directory, and then either rename the certificate to the default filename `private_key.pem` or change the value of the `proxy.config.admin.ssl_cert_file` variable in the `records.config` file to specify the file name of the certificate.

Enabling SSL

After you have obtained an SSL certificate, you can enable SSL by using the Traffic Manager UI or by editing a configuration file manually. Both procedures are provided below.

▼ To enable SSL from Traffic Manager:

1. From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).
2. On the **Configure** tab, click the **Security** button.

   The Security page opens displaying the **Control Access to the Traffic Server Manager** section at the top of the page.
3. Click the **SSL: On** button to enable SSL.

Note The SSL button displays only if you have obtained an SSL certificate and have copied it to Traffic Server's `config` directory.

4. Click the **Make These Changes** button.

▼ To enable SSL manually:

1 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

2 Edit the following variable:

Variable	Description
proxy.config.admin.use_ssl	Set this variable to 1 to enable SSL.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

Accessing Traffic Manager using SSL

To access the Traffic Manager UI from your browser using SSL, use the `https` command as shown below:

```
https://nodename:adminport
```

where *`nodename`* is the hostname of the Traffic Server node and *`adminport`* is the port number assigned to the Traffic Manager port (the default port number is 8081).

Configuring SOCKS firewall integration

SOCKS is commonly used as a network firewall that allows hosts behind a SOCKS server to gain full access to the Internet and prevents unauthorized access from the Internet to hosts inside the firewall.

Figure 17 illustrates how Traffic Server integrates into a SOCKS firewall.

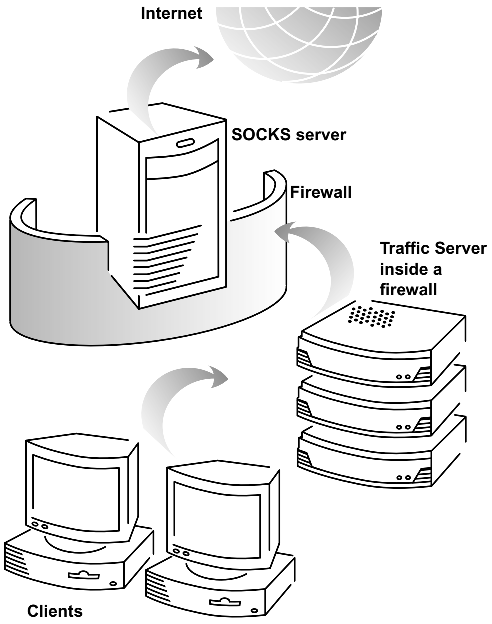


Figure 17 The Traffic Server inside a firewall using a SOCKS server

When Traffic Server receives a request for content that is not in the cache or is stale, it must request the content from the origin server. In a SOCKS configuration, instead of accessing the origin server directly, Traffic Server goes through the SOCKS Server. The SOCKS server authorizes communication between Traffic Server and the origin server, then relays the data to the origin server. The origin server then sends the content back to Traffic Server through the SOCKS server. Traffic Server caches the content and sends it to the client.

Setting SOCKS configuration options

To configure your Traffic Server to use a SOCKS firewall, you must:

✔ Enable the SOCKS option

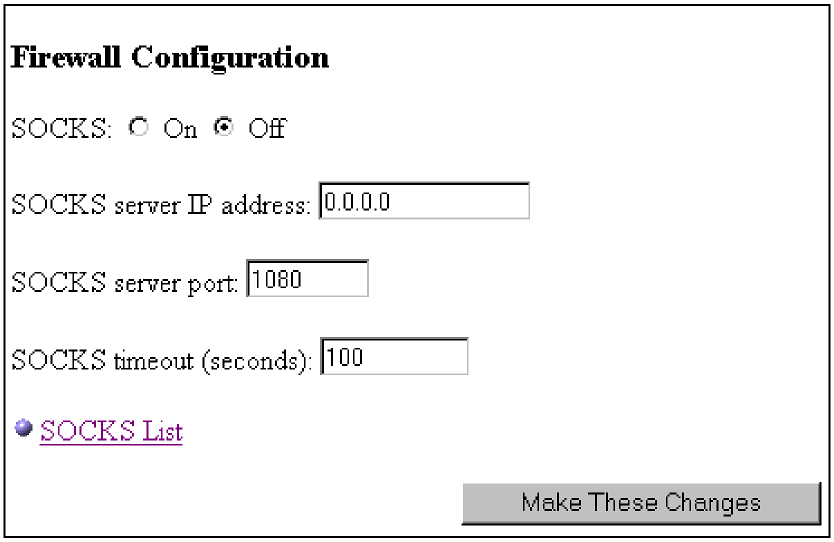
✔ Specify the IP address of your SOCKS server and the communication port

As an optional configuration step, you can specify the IP addresses of any origin servers that you want Traffic Server to access directly without going through the SOCKS server.

You can set SOCKS configuration options by using the Traffic Manager UI or by editing configuration files manually. Both procedures are provided below.

▼ To set SOCKS options from Traffic Manager:

1. From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).
2. On the **Configure** tab, click the **Security** button.
3. Scroll to the **Firewall Configuration** section (shown below).



Firewall Configuration

SOCKS: ○ On ● Off

SOCKS server IP address: 0.0.0.0

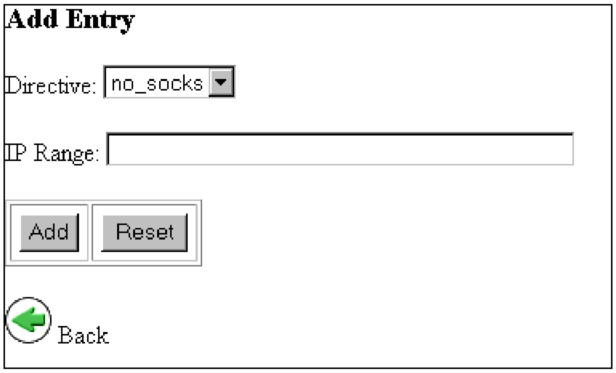
SOCKS server port: 1080

SOCKS timeout (seconds): 100

● SOCKS List

4. Select **SOCKS: On** to enable the SOCKS option.
5. In the **SOCKS server IP address** field, enter the IP address of your SOCKS server.
6. In the **SOCKS server port**, enter the port through which Traffic Server communicates with the SOCKS server.
7. In the **SOCKS timeout** field, enter the number of seconds the Traffic Server must wait for the SOCKS server to respond before dropping the connection.

   The default value is 100 seconds.
8. Click the **SOCKS List** link to specify the IP address of any origin server that you want to access directly without going through the SOCKS server.
9. On the Socks List page, click the **Add Entry** button to open the Add Entry dialog box (shown below).



Add Entry

Directive: no_socks

IP Range:

Currently, the only selection in the **Directive** drop-down list box is `no_socks`. This specifies that Traffic Server will access the origin server or group of servers listed in the **IP Range** field directly. Connections do *not* go through the SOCKS server.

10 In the **IP Range** field, specify a single IP address or range of IP addresses of the origin servers to which you want Traffic Server to connect directly.

11 Click the **Add** button.

12 Click the **Make These Changes** button.

▼ To set SOCKS options manually:

1 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

2 Edit the following variables:

Variable	Description
proxy.config.socks.socks_needed	Set this variable to 1 to enable SOCKS.
proxy.config.socks.socks_server_ip_str	Specify the IP address of the SOCKS server.
proxy.config.socks.socks_server_port	Specify the port used to communicate with the SOCKS server.
proxy.config.socks.socks_timeout	Specify the number of seconds the Traffic Server must wait for the SOCKS server to respond before dropping the connection.

3 Save and close the `records.config` file.

4 In a text editor, open the `socks.config` file located in Traffic Server's `config` directory.

5 Enter a line in the file specifying the IP addresses or IP address range of the origin servers that you want Traffic Server to access directly. Use the following format:

```
no_socks IPaddresses or IPaddress range
```

where *`IPaddresses or IPaddress range`* is a comma separated list of the IP addresses or IP address ranges associated with the origin servers you want Traffic Server to access directly.

6 Save and close the `socks.config` file.

7 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

8 Run the command `traffic_line -x` to apply the configuration changes.

Configuring DNS server selection (split DNS)

You can configure Traffic Server to use multiple DNS servers depending on your security requirements. For example, you can configure Traffic Server to look to one set of DNS servers to resolve hostnames on your internal network, while allowing DNS servers outside of the firewall to resolve hosts on the internet. This maintains the security of your intranet, while continuing to provide direct access to sites outside your organization.

You specify the rules for performing DNS server selection (also called *split DNS*) in the `splitdns.config` file. Traffic Server enables you to specify this selection based on the destination domain, the destination host, or a URL regular expression.

▼ To configure DNS server selection:

1. In a text editor, open the `records.config` file located in Traffic Server's `config` directory.
2. Edit the following variable:

Variable	Description
proxy.process.dns.splitDNS.enabled	Set this variable to 1 to enable split DNS.

3. Save and close the `records.config` file.
4. In a text editor, open the `splitdns.config` file located in Traffic Server's `config` directory.
5. Add rules to the `splitdns.config` file.

   For information about the format of the `splitdns.config` file, see *page 294*.
6. Save and close the `splitdns.config` file.
7. In *UNIX*, make Traffic Server's `bin` directory your working directory.

   In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).
8. Run the command `traffic_line -x` to apply the configuration changes.

Configuring LDAP-based proxy authentication

Traffic Server enables you to leverage existing directory services by supporting asynchronous match and bind requests to LDAP servers, thereby supporting policies that require users to log in and be authenticated by the proxy. You can use results from this authentication to enforce rules related to whether users have the authority to go out onto the Internet.

Traffic Server uses a local database to improve the performance of LDAP authentications and, upon completion, logs successfully authenticated users.

▼ To configure LDAP-based proxy authentication:

1 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

2 Edit the following variables:

Variable	Description
proxy.config.ldap.auth.enabled	Set this variable to 1 to enable LDAP-based proxy authentication.
proxy.config.ldap.proc.ldap.server.name	Set this variable to specify the name of the LDAP server.
proxy.config.ldap.proc.ldap.server.port	Set this variable to specify the LDAP port number. The default port number is 389.
proxy.config.ldap.proc.ldap.base.dn	Set this variable to specify the name of the base Distinguished Name (DN). Obtain this value from your LDAP administrator. You must specify a correct base DN otherwise LDAP authentication will fail to operate.

3 Save and close the `records.config` file.

4 Restart Traffic Server.

Configuring LDAP Authentication Bypass

You can enable Traffic Server clients to access specific sites on the Internet without being authenticated by the LDAP server.

▼ To enable clients to access specific sites without LDAP authentication:

1 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

2 Edit the following variables:

Variable	Description
proxy.config.ldap.auth.bypass.enabled	Set this variable to 1 to enable LDAP authentication bypass.
proxy.config.ldap.auth.multiple.ldap_servers.enabled	Set this variable to 1 to allow the sites specified in the `ldapsrvr.config` file to bypass authentication.

3 Save and close the `records.config` file.

4 In a text editor, open the `ldapsrvr.config` file.

5 Add bypass sites to the `ldapsrvr.config` file.

For information about the format of the `ldapsrvr.config` file, refer to *page 241*.

6 Save and close the `ldapsrvr.config` file.

7 Restart Traffic Server.

Using SSL Termination

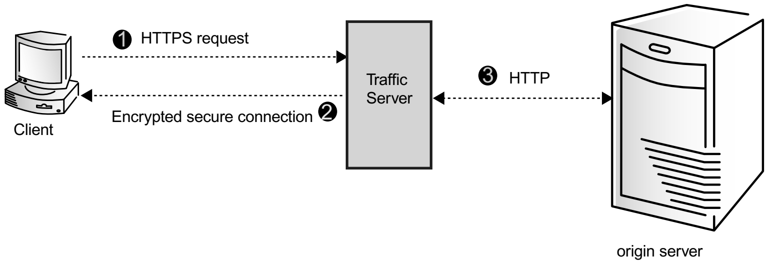
Traffic Server's SSL termination option enables you to secure connections in reverse proxy mode between a client and a Traffic Server and/or Traffic Server and an origin server.

The following sections describe how to enable and configure the SSL termination option:

- ✔ To enable and configure SSL termination for client/Traffic Server connections, follow the procedures in *Client and Traffic Server connections* below.
- ✔ To enable and configure SSL termination for Traffic Server/origin server connections, refer to *Traffic Server and origin server connections, on page 150*.
- ✔ To enable and configure SSL termination for both client/Traffic Server and Traffic Server/origin server connections, follow the procedures in both *Client and Traffic Server connections* below, and *Traffic Server and origin server connections, on page 150*.

Client and Traffic Server connections

Figure 18 illustrates communication between a client and Traffic Server, and between Traffic Server and an origin server when the SSL termination option is enabled and configured for client/Traffic Server connections only.



❶ The client sends an HTTPS request for content. Traffic Server receives the request and performs the SSL handshake to authenticate the client (depending on the authentication options configured) and to determine the encryption method to be used.
If the client is allowed access, Traffic Server checks its cache for the requested content.

❷ If the request is a cache hit and the content is fresh, Traffic Server encrypts the content and sends it to the client, where it is decrypted (using the method determined during the handshake) and displayed.

❸ If the request is a cache miss or is stale, Traffic Server communicates with the origin Server via HTTP and obtains the plain text version of the content. Traffic Server saves the plain text version of the content in its cache, and then encrypts the content and sends it to the client, where it is decrypted and displayed.

Figure 18 Client and Traffic Server communication using SSL termination

To configure Traffic Server to use the SSL termination option for client/Traffic Server connections:

✔ Obtain and install an SSL *server* certificate from a recognized certificate authority (such as VeriSign). The SSL server certificate contains information that allows the client to authenticate Traffic Server and exchange secret encryption keys.

✔ Set configuration variables in the `records.config` file to:

- ✗ Enable the SSL termination option
- ✗ Set the port number used for SSL communication
- ✗ Specify the filename and location of the server certificate
- ✗ Configure the use of client certificates (optional)

  Client certificates are located on the client. If you configure Traffic Server to require client certificates, Traffic Server verifies the client certificate during the SSL handshake to authenticate the client. This authentication process is transparent to the user. If you configure Traffic Server to *not* require client certificates, access to Traffic Server is managed through access control lists and other Traffic Server options that have been set (for example rules in the `ip_allow.config` file and LDAP-based proxy authentication).

- ✗ Specify the file name and location of the Traffic Server's private key (if the private key is not located in the server certificate file)

  Traffic Server uses its private key during the SSL handshake to decrypt the session encryption keys. The private key must be stored and protected against theft.

- ✗ Configure the use of certification authorities (CAs) *- Optional*

  CAs provide added security when using client certificates by verifying the identity of the person requesting a certificate.

▼ **To set SSL termination configuration variables for client/Traffic Server connections:**

1 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

2 Edit the following variables in the `SSL Termination` section of the file:

Variable	Description
proxy.config.ssl.enabled	Set this variable to 1 to enable the SSL termination option.
proxy.config.ssl.server_port	Set this variable to specify the port used for SSL communication. The default port is 443.
proxy.config.ssl.client.certification_level	Set this variable to one of the following values: ▮ 0 specifies that no client certificates are required. Traffic Server does not verify client certificates during the SSL handshake. Access to Traffic Server depends on Traffic Server configuration options (such as access control lists). ▮ 1 specifies that client certificates are optional. If a client has a certificate, the certificate is validated. If the client does not have a certificate, the client is still allowed access to Traffic Server unless access is denied through other Traffic Server configuration options. ▮ 2 specifies that client certificates are required. The client must be authenticated during the SSL handshake. Clients without a certificate are not allowed to access Traffic Server.

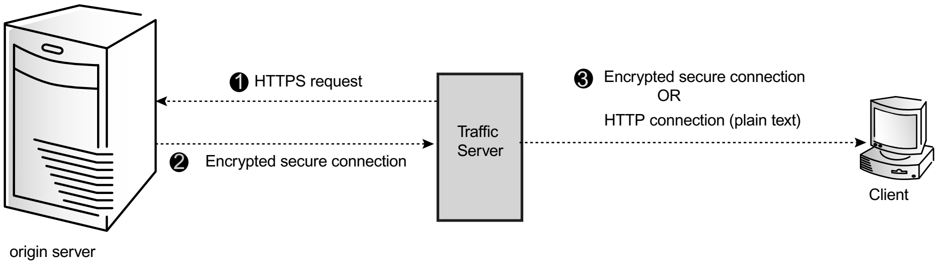
Variable	Description
proxy.config.ssl.server.cert.filename	Set this variable to specify the file name of Traffic Server's SSL server certificate. Traffic Server provides a demo server certificate called `server.pem`. You can use this certificate to verify that the SSL feature is working. If you are using multiple server certificates, set this variable to specify the default file name.
proxy.config.ssl.server.cert.path	Set this variable to specify the location of Traffic Server's SSL server certificate. The default directory is Traffic Server's `config` directory.
proxy.config.ssl.server.private_key.filename	Set this variable to specify the file name of Traffic Server's private key. Change this variable only if the private key is not located in the Traffic Server's SSL server certificate file.
proxy.config.ssl.server.private_key.path	Set this variable to specify the location of the Traffic Server's private key. Change this variable only if the private key is not located in the Traffic Server's SSL server certificate file.
proxy.config.ssl.CA.cert.filename	Specify the file name of the certificate authority that client certificates will be verified against.The default value is NULL.
proxy.config.ssl.CA.cert.path	Specify the location of the certificate authority file that client certificates will be verified against.The default value is NULL.

3 Save and close the `records.config` file.

4 Restart Traffic Server.

Traffic Server and origin server connections

Figure 18 illustrates communication between Traffic Server and an origin server when the SSL termination option is enabled for Traffic Server /origin server connections.


❶ HTTPS request
Traffic Server
❸ Encrypted secure connection
OR
HTTP connection (plain text)
❷ Encrypted secure connection
Client
origin server

❶ If a client request is a cache miss or is stale, Traffic Server sends an HTTPS request for the content to the origin server. The origin server receives the request and performs the SSL handshake to authenticate Traffic Server and to determine the encryption method to be used.

❷ If Traffic Server is allowed access, the origin server encrypts the content and sends it to Traffic Server, where it is decrypted (using the method determined during the handshake) and the plain text version of the content saved in the cache.

❸ If SSL termination is enabled for client /Traffic Server connections, Traffic Server re-encrypts the content and sends it to the client via HTTPS, where it is decrypted and displayed.
If SSL termination is not enabled for client/Traffic Server connections, Traffic Server sends the plain text version of the content to the client via HTTP.

Figure 19 Traffic Server and origin server communication using SSL termination

To configure Traffic Server to use the SSL termination option for Traffic Server and origin server connections:

✔ Obtain and install an SSL *client* certificate from a recognized certificate authority (such as VeriSign). The SSL client certificate contains information that allows Traffic Server to authenticate the origin server and exchange secret encryption keys.

The client certificate is optional. If you do not install an SSL client certificate, you must use a certification authority (CA).

✔ Set configuration variables in the `records.config` file to:

✗ Enable the SSL termination option

✗ Set the port number used for SSL communication

✗ Specify the filename and location of the SSL client certificate (if you choose to use a client certificate)

✗ Specify the file name and location of the Traffic Server's private key (if the private key is not located in the client certificate file)

Traffic Server uses its private key during the SSL handshake to decrypt the session encryption keys. The private key must be stored and protected against theft.

✗ Configure the use of CAs. You must use a CA if you choose not to use a client certificate, otherwise, security on your system may be compromised.

CAs allows the Traffic Server that is acting as a client to verify the identity of the server with which it is communicating and to exchange secret encryption keys.

▼ To set SSL termination configuration variables for Traffic Server/origin server connections:

1 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

2 Edit the following variables in the `SSL Termination` section of the file:

Variable	Description
proxy.config.ssl.auth.enabled	Set this variable to 1 to enable the SSL termination option.
proxy.config.ssl.server_port	Set this variable to specify the port used for SSL communication. The default port is 443.
proxy.config.ssl.client.verify.server	Set this option to 1 to require Traffic Server to verify the origin server certificate with the CA.
proxy.config.ssl.client.cert.filename	If you have installed an SSL client certificate on Traffic Server, set this variable to specify the file name of client certificate.
proxy.config.ssl.client.cert.path	If you have installed an SSL client certificate on Traffic Server, set this variable to specify the location of the client certificate. The default directory is Traffic Server's `config` directory.
proxy.config.ssl.client.private_key.filename	Set this variable to specify the file name of Traffic Server's private key. Change this variable only if the private key is not located in the Traffic Server's SSL client certificate file.
proxy.config.ssl.client.private_key.path	Set this variable to specify the location of the Traffic Server's private key. Change this variable only if the private key is not located in the SSL client certificate file.
proxy.config.ssl.client.CA.cert.filename	Specify the file name of the certificate authority against which the origin server will be verified.The default value is NULL.
proxy.config.ssl.client.CA.cert.path	Specify the location of the certificate authority file against which the origin server will be verified.The default value is NULL.

3 Save and close the `records.config` file.

4 Restart Traffic Server.

Chapter 12

Working with Log Files

Traffic Server generates log files that contain information about every request it receives and every errors it detects.

This chapter discusses the following topics:

Understanding Traffic Server log files

Traffic Server records information about each transaction (or request) that it processes and every error that it detects in log files. Traffic Server keeps three types of log files:

- ✔ *System log files* record system information, which includes messages about the state of Traffic Server and any errors or warnings that it produces. This kind of information might include a note that event log files were rolled, a warning that cluster communication timed out, or an error indicating that Traffic Server was restarted. (Traffic Server posts alarms signifying error conditions on Traffic Server's **Dashboard**; see *The Dashboard page, on page 180* for details.)

  In *UNIX*, all system information messages are logged with the system-wide logging facility `syslog` under the daemon facility. The `syslog.conf` configuration file (stored in the `/etc` directory) specifies where these messages are logged. A typical location is `/var/adm/messages`.

  Since the `syslog` process works on a system-wide basis, it serves as the single repository for messages from all Traffic Server processes, including `traffic_server`, `traffic_manager`, and `traffic_cop`.

  In *Windows*, system information messages from the `traffic_server` and `traffic_manager` processes are logged in the application log in the Windows Event Log. To view the application log, click the Windows **Start** button, then select **Programs/Administrative Tools (Common)/Event Viewer**. From the **Log** menu, select **Application**. Messages from the `traffic_cop` process are logged to the `cop.log` file (located in the Traffic Server installation directory).

  System information logs observe a static format. Each log entry in the log contains information about the date and time the error was logged, the hostname of the Traffic Server that reported the error, and a description of the error or warning.

  See *Appendix F, Traffic Server Error Messages* for a list of the system information messages that Traffic Server logs.

- ✔ *Error log files* record information about why a particular transaction was in error.
- ✔ *Event log files* record information about the state of each transaction that Traffic Server processes.

Understanding event log files

Event log files record information about every request that Traffic Server processes. By analyzing the log files, you can determine how many people use the Traffic Server cache, how much information each person requested, what pages are most popular, and so on.

Traffic Server supports several standard log file formats, such as Squid and Netscape, and user-defined custom formats. You can analyze the standard format log files with off-the-shelf analysis packages. To help with log file analysis, you can separate log files so that they contain information specific to protocol or hosts. You can also configure Traffic Server to roll log files automatically at specific intervals during the day.

The following sections describes the Traffic Server logging system features and discusses how to:

✔ **Manage your event log files**

You can choose a central location for storing log files, set how much disk space to use for log files, and set how and when to roll log files. See *Managing event log files, on page 156*.

✔ **Choose different event log file formats**

You can choose which standard or custom log file formats you want to use for traffic analysis. See *Choosing event log file formats, on page 158*.

✔ **Rotate event log files automatically**

You can configure Traffic Server to rotate event log files at specific intervals during the day so that you can identify and manipulate log files that are no longer active. See *Rolling event log files, on page 167*.

✔ **Separate log files according to protocols and hosts**

You can configure Traffic Server to create separate log files for NNTP, ICP, and HTTP/FTP transactions. You can also configure Traffic Server to generate separate log files for different protocols based on the host. See *Splitting event log files, on page 170*.

✔ **Collate log files from different Traffic Server nodes**

You can designate one or more nodes on the network to serve as log collation servers. These servers, which may either be stand-alone or part of Traffic Server, enable you to keep all logged information in well-defined locations. See *Collating event log files, on page 173*.

✔ **View statistics about the logging system**

Traffic Server provides statistics about the logging system. You can access the statistics through the Traffic Manager UI or through Traffic Line. See *Viewing logging statistics, on page 178*.

Managing event log files

You can manage your event log files and control where they are located, how much space they can consume, and how low disk space in the logging directory is handled.

Choosing the logging directory

By default, Traffic Server writes all event log files in the `logs` directory, which is located in the directory where you installed Traffic Server. To use a different directory, see *Setting log file management options* below.

Controlling logging space

Traffic Server allows you to control the amount of disk space that the logging directory can consume. This allows the system to operate smoothly within a specified space window for a long period of time.

After you establish a space limit, Traffic Server continues to monitor the space in the logging directory. When the free space dwindles to the headroom limit (see *Setting log file management options, on page 156*), it enters a low space state and takes the following actions:

- ✔ If the autodelete option (discussed in *Rolling event log files, on page 167*) is *enabled*, Traffic Server identifies previously rolled log files (log files with a `.old` extension) and starts deleting files one by one—beginning with the oldest file—until it emerges from the low state. Traffic Server logs a record of all files it deletes in the system error log.
- ✔ If the autodelete option is *disabled*, or there are not enough old log files to delete for the system to emerge from its low space state, Traffic Server issues a warning and continues logging until space is exhausted, at which point it stops event logging. Traffic Server resumes event logging when enough space becomes available for it to exit its low space state. You can make space available either by removing files from the logging directory manually or by explicitly increasing the logging space limit.

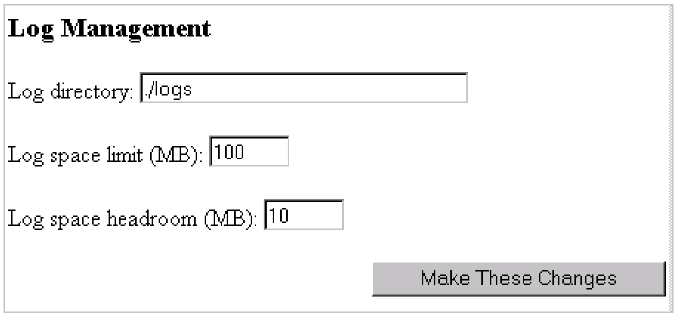
You can run a `cron` script in conjunction with Traffic Server to automatically remove old log files from the logging directory (before Traffic Server enters the low space state) and relocate them to a temporary partition. Once the files are relocated, you can run a variety of log analysis scripts on them, after which you can compress the logs and move them to an archive location or simply delete them.

Setting log file management options

You can set log management options by using the Traffic Manager UI or by editing a configuration file manually. Both procedures are provided below.

▼ To set log management options from Traffic Manager:

1. From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).
2. On the **Configure** tab, click the **Logging** button.
3. Scroll to the **Log Management** section of the **Logging** page (shown below).



Log Management

Log directory: ./logs

Log space limit (MB): 100

Log space headroom (MB): 10

4. In the **Log directory** field, enter the name and path of the directory in which you want to store event log files. The default directory is `logs` located in Traffic Server's installation directory.
5. In the **Log space limit** field, enter the maximum amount of space you want to allocate to the logging directory.
6. In the **Log space headroom** field, enter the tolerance for the log space limit. If the **Autodelete rolled log files when space is low** option is enabled in the **Log File Rolling** section of the **Logging** page, auto deletion is triggered when the amount of free space available in the logging directory is less than the headroom.
7. Click the **Make These Changes** button.

▼ To set log management options manually:

1. In a text editor, open the `records.config` file located in the `config` directory.
2. Edit the following variables:

Variable	Description
proxy.config.log2.logfile_dir	Specify the name and path of the directory in which you want to store event log files. The default is `logs` located in the directory where you installed Traffic Server.
proxy.config.log2.max_space_mb_for_logs	Enter the maximum amount of space you want to allocate to the logging directory.
proxy.config.log2.max_space_mb_headroom	Enter the tolerance for the log space limit.

3. Save and close the `records.config` file.
4. In *UNIX*, make Traffic Server's `bin` directory your working directory.

   In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).
5. Run the command `traffic_line -x` to apply the configuration changes.

Choosing event log file formats

Traffic Server supports the following log file formats:

- ✔ *Standard formats*, such as Squid or Netscape (see *Using standard formats* below).
- ✔ *Custom formats: traditional or XML-based*. If you choose to use the XML-based format, you can also create *summary log files*, which reduce the size of the generated log files significantly. (See *Using custom formats, on page 161*.)

In addition to the standard and custom log file formats, you must choose whether to save log files in *binary* or *ASCII*. See *Choosing binary or ASCII, on page 165*.

Important Event log files consume a large amount of disk space. Creating log entries in multiple formats at the same time can consume disk resources very quickly and adversely impact Traffic Server performance.

Using standard formats

The standard log formats include Squid, Netscape Common, Netscape extended, and Netscape Extended-2.

The standard log file formats can be analyzed with a wide variety of off-the-shelf log-analysis packages. You should use one of the standard event log formats unless you need information that these formats do not provide. See *Using custom formats, on page 161*.

When using standard log file formats, Traffic Server can make certain optimizations in collecting and formatting the data since it knows what data will be needed, and in what format. It is faster to use the standard log file formats instead of the custom log file formats. In addition, the Squid format is faster to process than the Netscape formats. By default, Traffic Server is configured to use the Squid log file format only.

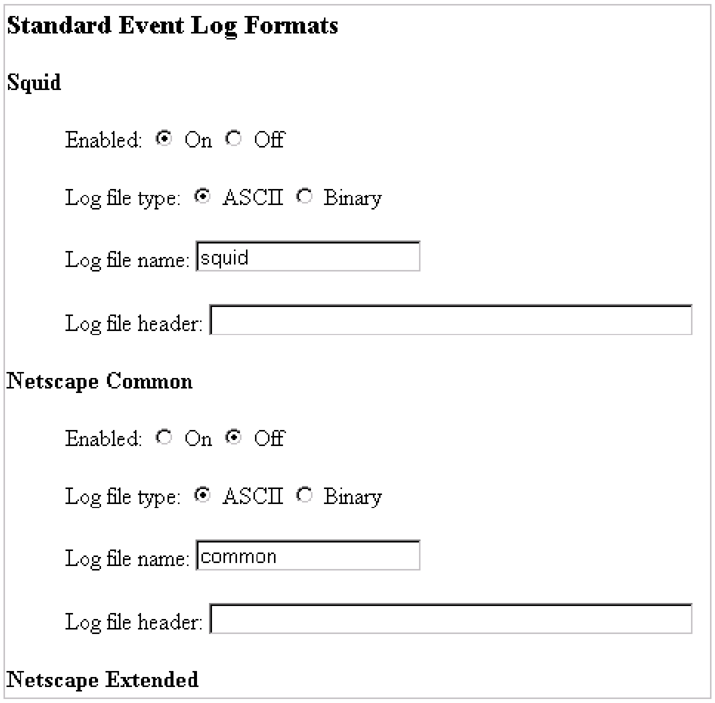
Setting standard log file format options

You can set standard log file format options by using the Traffic Manager or by editing a configuration file manually. Both procedures are provided below.

▼ To select a standard event log file format from Traffic Manager:

1. From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).
2. On the **Configure** tab, click the **Logging** button.

3 Scroll to the **Standard Event Log Formats** section of the **Logging** page (shown below).



Standard Event Log Formats

Squid

Enabled: ● On ○ Off

Log file type: ● ASCII ○ Binary

Log file name: squid

Log file header:

Netscape Common

Enabled: ○ On ● Off

Log file type: ● ASCII ○ Binary

Log file name: common

Log file header:

Netscape Extended

4 Click the **Enabled:On** button for the format you want to use.

5 Select the log file type (ASCII or binary).

6 In the **Log file name** field, enter the name you want to use for the event log file.

7 In the **Log file header** field, enter a text header that will display at the top of the event log file. Leave this field blank if you do not want to use a text header.

8 Click the **Make These Changes** button.

▼ To select a standard event log file format manually:

1 In a text editor, open the `records.config` file located in the `config` directory.

2 To use the Squid format, edit the following variables:

Variable	Description
proxy.config.log2.squid_log_enabled	Set this variable to 1 to enable the Squid log file format.
proxy.config.log2.squid_log_is_ascii	Set this variable to 1 to enable ASCII mode. Set this variable to 0 to enable binary mode.
proxy.config.log2.squid_log_name	Enter the name you want to use for Squid event log files. The default is `squid`.
proxy.config.log2.squid_log_header	Enter the header text you want to display at the top of the Squid log files.

3 To use the Netscape Common format, edit the following variables:

Variable	Description
proxy.config.log2.common_log_enabled	Set this variable to 1 to enable the Netscape Common log file format.
proxy.config.log2.common_log_is_ascii	Set this variable to 1 to enable ASCII mode. Set this variable to 0 to enable binary mode.
proxy.config.log2.common_log_name	Enter the name you want to use for Netscape Common event log files. The default is `common`.
proxy.config.log2.common_log_header	Enter the header text you want to display at the top of the Netscape Common log files.

4 To use the Netscape Extended format, edit the following variables:

Variable	Description
proxy.config.log2.extended_log_enabled	Set this variable to 1 to enable the Netscape Extended log file format.
proxy.config.log2.extended_log_is_ascii	Set this variable to 1 to enable ASCII mode. Set this variable to 0 to enable binary mode.
proxy.config.log2.extended_log_name	Enter the name you want to use for Netscape Extended event log files. The default is `extended`.
proxy.config.log2.extended_log_header	Enter the header text you want to display at the top of the Netscape Extended log files.

5 To use the Netscape Extended2 format, edit the following variables:

Variable	Description
proxy.config.log2.extended2_log_enabled	Set this variable to 1 to enable the Netscape Extended2 log file format.
proxy.config.log2.extended2_log_is_ascii	Set this variable to 1 to enable ASCII mode. Set this variable to 0 to enable binary mode.
proxy.config.log2.extended2_log_name	Enter the name you want to use for Netscape Extended2 event log files. The default is `extended2`.
proxy.config.log2.extended2_log_header	Enter the header text you want to display at the top of the Netscape Extended2 log files.

6 Save and close the `records.config` file.

7 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

8 Run the command `traffic_line -x` to apply the configuration changes.

Using custom formats

In addition to the standard log file formats, Traffic Server also supports two models for specifying custom log files:

- ✔ *Traditional* (simple)
- ✔ *XML-based* (highly configurable)

  Using the XML-based format, you can also create *summary* log files.

Create a custom log specification if you need data for analysis that is not available in the standard formats. You can decide what information to record for each Traffic Server transaction, and using the XML-based custom format, you can create filters to define which transactions to log.

Using traditional custom formats

To create traditional custom log files, you must enable the traditional custom log format option and edit Traffic Server's traditional log configuration file (`logs.config`). You must specify the information you want to display in your log files by entering printf-style format strings.

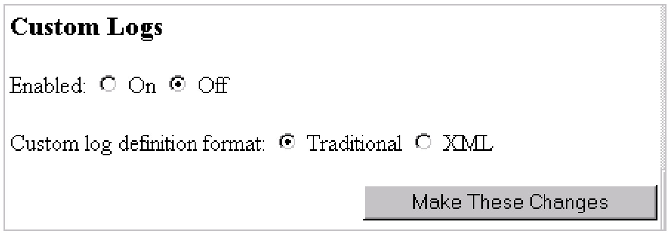
You can create as many custom log file formats as necessary. However, log files consume space quickly, so do not create more than you really need.

▼ To create traditional custom log formats:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Logging** button.

3 Scroll to the **Custom Logs** section of the **Logging** page (shown below).



Custom Logs

Enabled: ○ On ● Off

Custom log definition format: ● Traditional ○ XML

4 Select the **Enabled: On** button.

5 Select the **Traditional** button.

6 Click the **Make These Changes** button, then exit Traffic Manager.

7 In a text editor, open the `logs.config` file.

8 To add a traditional custom log specification, enter a line in the `logs.config` file with the following format:

```
format:enable_flag:format_id:format_name:string:file_name:type:header
```

Note Each line in the `logs.config` file produces one file (if the format is active).

Specify values for the fields:

Field	Description
format	All lines must begin with the word format.
enable_flag	This field enables or disables transaction logging using this format. You can enter one of the following: ▮ enabled ▮ disabled If you enter disabled, you have defined a custom log file format but are not using it.
format_id	Use a unique integer to identify each of your custom log formats.
format_name	Enter a name for this custom format.
string	Enter a printf-style format string specifying the field symbols to be displayed and how they should look in ASCII. For a list of the available field symbols and their meanings, refer to *Appendix E, Event Logging Formats*.
file_name	Enter a name you want to use for the log file created with this format.
type	Enter one of the following: ▮ ASCII ▮ BINARY
header	If you want your custom log file to have header text, enter it here.

Example The following example custom log definition produces a log file that records the client host IP address, the client request universal resource identifier, and the proxy response status code:

```
format:enabled:1:minimal:%<chi> / %<cqu> / %<pssc>:minimalist:ASCII:myheader
```

For more information about configuring the logs.config file, see *logs.config, on page 242*.

9 Save and close the logs.config file.

10 In *UNIX*, make Traffic Server's bin directory your working directory.

In *Windows*, open a command prompt window and change to the bin directory (located in the Traffic Server installation directory).

11 Run the command traffic_line -x to apply the configuration changes.

Using XML-based custom formats

Traffic Server supports a broad range of logging formats, including both standard and traditional custom formats. However, associating these formats to specific servers, collation hosts, and log files is rather coarse-grained. For example, when you enable collation, it is generally enabled for all formats and only to a single collation server.

Traffic Server's XML-based custom log specification is more flexible, enabling you to institute much more control over the type of information recorded in your log files.

The heart of the XML-based custom logging feature is an XML-based logging configuration file (logs_xml.config) that enables you to create very modular descriptions of logging objects. The logs_xml.config file uses three types of objects to create custom log files:

✔ The LogFormat object defines the content of the log file using printf-style format strings.

✔ The LogFilter object defines a filter so that you include or exclude certain information from the log file.

✔ The `LogObject` object specifies all the information needed to produce a log file. For example:

- ✗ The name of the log file (required).
- ✗ The format to be used (required). This can be a standard format (Squid or Netscape) or a previously defined custom format (a previously defined `LogFormat` object).
- ✗ The file mode (ASCII, Binary, or ASCII_PIPE). The default is ASCII.

  The ASCII_PIPE mode writes an XML-based custom log file to a pipe so that the logging data is sent to a buffer in memory. Other processes can then read the data using standard I/O functions. The advantage of using this option is that Traffic Server does not have to write to disk, freeing disk space for other tasks.
- ✗ Any filters you want to use (previously defined `LogFilter` objects).
- ✗ The collation servers that are to receive the flog files.
- ✗ The protocols you want to log (if the protocols tag is used, Traffic Server will only log transactions from the protocols listed, otherwise, all transactions for all protocols are logged).
- ✗ The origin servers you want to log (if the servers tag is used, Traffic Server will only log transactions from the origin servers listed, otherwise, transactions from all origin servers are logged).

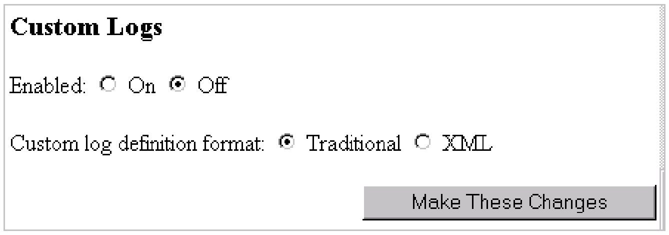
Note To generate a custom log file, you must specify at least one `LogObject` definition. One log file is produced for each `LogObject` definition.

▼ To generate XML-based custom log files:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Logging** button.

3 Scroll to the **Custom Logs** section of the **Logging** page (shown below).



Custom Logs

Enabled: ○ On ● Off

Custom log definition format: ● Traditional ○ XML

4 Select the **Enabled: On** button.

5 Select the **XML** button.

6 Click the **Make These Changes** button, then exit Traffic Manager.

7 In a text editor, open the `records.config` file located in Traffic Server's `config` directory.

8 Search for the following string:

```
CONFIG proxy.config.log2.xml_config_file STRING
```

9 After `STRING`, enter the name of the XML-based logging configuration file.

By default, this is `logs_xml.config`, located in the `config` directory.

10. Save and close the `records.config` file.
11. In a text editor, open the `logs_xml.config` file.
12. Add `LogFormat`, `LogFilter`, and `LogObject` specifications to the configuration file.

    For detailed information about the `logs_xml.config` file and associated object specifications, see *logs_xml.config, on page 245*.
13. Save and close the `logs_xml.config` file.
14. In *UNIX*, make Traffic Server's `bin` directory your working directory.

    In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).
15. Run the command `traffic_line -x` to apply the configuration changes.

Creating summary log files

Traffic Server performs several hundred operations per second, therefore, event log files can grow quickly to large sizes. Using SQL-like aggregate operators, you can configure Traffic Server to create summary log files that summarize a set of log entries over a specified period of time. This can significantly reduce the size of the log files generated.

You generate a summary log file by creating a `LogFormat` object in the XML-based logging configuration file (`logs_xml.config`) using the following SQL-like aggregate operators:

✔ COUNT

✔ SUM

✔ AVERAGE

✔ FIRST

✔ LAST

You can apply each of these operators to specific fields, requesting it to operate over a specified interval.

Summary logs represent a trade-off between convenience and information granularity. Since you must specify a time interval during which only a single record is generated, you will necessarily be losing a certain amount of information. If you want the convenience of summary logs, but also need the detail of a conventional log file, consider creating and enabling two custom log formats—one using aggregate operators and the other not.

▼ To create a summary log file:

1. In a text editor, open the `logs_xml.config` file.
2. Define the format of the log file as follows:

```
<LogFormat>
  <Name = "summary"/>
  <Format = "%<operator(field)> : %<operator(field)>"/>
  <Interval = "n"/>
</Format>
```

where:
`operator` is one of the five aggregate operators (COUNT, SUM, AVERAGE, FIRST, LAST). You can specify more than one operator in the format line.

`field` is the logging field that you want to aggregate.

`n` is the interval in seconds between summary log entries.

For more information, see *logs_xml.config, on page 245*.

For example, the following format generates one entry every 10 seconds, with each entry summarizing the timestamp of the last entry of the interval, a count of the number of entries seen within that 10-second interval, and the sum of all bytes sent to the client.

```
<LogFormat>
  <Name = "summary"/>
  <Format = "%<LAST(cqts)> : %<COUNT(*)> : %<SUM(psql)>"/>
  <Interval = "10"/>
</Format>
```

Important You cannot create a format specification that contains both aggregate operators and regular fields. For example, the following specification would be invalid:

```
<Format = "%<LAST(cqts)> : %<COUNT(*)> : %<SUM(psql)> : %<cqu>"/>
```

3. Define a `LogObject` that uses this format.
4. Save and close the configuration file.
5. In *UNIX*, make Traffic Server's `bin` directory your working directory.

   In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).
6. Run the command `traffic_line -x` to apply the configuration changes.

Choosing binary or ASCII

You can configure the Traffic Server to create event log files in either of the following:

✔ ASCII - these files are human readable and can be processed using standard, off-the-shelf log analysis tools. However, Traffic Server must perform additional processing to create the files in ASCII, resulting is a slight increase in overhead. Also, ASCII files tend to be larger than the equivalent binary files. ASCII log files have a `.log` file name extension by default.

✔ Binary - files in binary have the advantage of generating lower system overhead, as well as generally occupying less space on the disk depending on the type of information being logged. You must, however, use a converter application before you can read or analyze these files using standard tools. Binary log files use a `.blog` file name extension by default.

While binary log files typically require less disk space, this is not always the case. For example, the value 0 (zero) requires only one byte to store in ASCII, but requires four bytes when stored as a binary integer. On the other hand, if you define a custom format that logs IP addresses, a binary log file would only require four bytes of storage per 32-bit address. However, the same IP address stored in dot notation would require around 15 characters (bytes) in an ASCII log file.

For standard log formats, you select Binary or ASCCI in the **Standard Event Log Formats** section of the **Logging** page, refer to *Setting standard log file format options, on page 158*. For custom log formats, refer to *Using traditional custom formats, on page 161*.

Before selecting ASCII versus binary for your log files, consider the type of data that will be logged. Try logging for one day using ASCII and then one day using binary. Assuming that the number of requests is roughly the same for both days, you can calculate a rough metric comparing the two formats.

Note For XML-based custom log files, in addition to the ASCII and binary options, you can also write a log file to a pipe so that the logging data is sent to a buffer in memory. Other processes can then read the data using standard I/O functions. The advantage of using this option is that Traffic Server does not have to write to disk, freeing disk space for other tasks. In addition, writing to a pipe does not stop when logging space is exhausted because the pipe does not use disk space. Refer to *logs_xml.config, on page 245* for more information about the ASCII_PIPE option.

Using logcat to convert binary logs to ASCII

You must convert a binary log file to ASCII before you can analyze it using standard tools.

▼ To convert a binary log file to ASCII:

1 Change to the directory containing the binary log file.

2 Make sure that the `logcat` utility is in your path.

3 Enter the following command:

```
logcat [options] [input_filename...]
```

The following table describes the command-line options:

Option	Description
-o output_file	Specifies where the command output is directed.
-a	Automatically generates the output file name based on the input file name. If the input is from stdin, this option is ignored. For example: logcat -a squid-1.blog squid-2.blog squid-3.blog generates squid-1.log, squid-2.log, squid-3.log
-S	Attempts to transform the input to Squid format, if possible.
-C	Attempts to transform the input to Netscape Common format, if possible.
-E	Attempts to transform the input to Netscape Extended format, if possible.
-2	Attempt to transform the input to Netscape Extended-2 format, if possible.

Note Use only one of the following options at any given time: -S, -C, -E, or -2.

If no input files are specified, `logcat` reads from the standard input (`stdin`). If you do not specify an output file, `logcat` writes to the standard output (`stdout`).

For example, to convert a binary log file to an ASCII file, you can use the `logcat` command with either of the following options:

```
logcat binary_file > ascii_file
```

```
logcat -o ascii_file binary_file
```

The binary log file is not modified by this command.

Rolling event log files

Traffic Server provides automatic log file rolling. This means that at specific intervals during the day, Traffic Server closes its current set of log files and opens new log files.

Log file rolling offers the following benefits:

- ✔ It defines an interval over which log analysis can be performed
- ✔ It keeps any single log file from becoming too large and assists in keeping the logging system within the specified space limits
- ✔ It provides an easy way to identify files that are no longer being used so that an automated script can clean the logging directory and run log analysis programs

You should roll log files several times a day. Rolling every 6 hours is a good guideline to follow.

Rolled log file name format

Traffic Server provides a consistent name format for rolled log files that allows you to easily identify log files.

When Traffic Server rolls a log file, it saves and closes the old file and starts a new file. The old file is renamed to include:

- ✔ The hostname of the Traffic Server that generated the log file
- ✔ Two time stamps separated by a hyphen (`-`) that represent the earliest and latest time stamps for the entries contained in the rolled log file
- ✔ The suffix `.old` that makes it easy for automated scripts to find rolled log files

The time stamps have the following format:

```
%Y%M%D.%Hh%Mm%Ss-%Y%M%D.%Hh%Mm%Ss
```

where:

Code	Definition	Example
`%Y`	the year in four-digit format	2000
`%M`	the month in two-digit format, from 01-12	07
`%D`	the day in two-digit format, from 01-31	19
`%H`	the hour in two-digit format, from 00-23	21
`%M`	the minute in two-digit format, from 00-59	52
`%S`	the second in two-digit format, from 00-59	36

The following is an example of a rolled log file name:

```
squid.log.mymachine.20000912.12h00m00s-20000913.12h00m00s.old
```

When a log file is rolled, the log buffer might be partially full. If so, the first entry in the new log file will have a time stamp earlier than the time of rolling. When the new log file is rolled, its time stamp will be the time stamp of the first entry. For example, suppose logs are rolled every 3 hours, and the first rolled log file is:

Log file 1

```
squid.log.mymachine.19980912.12h00m00s-19980912.03h00m00s.old
```

If the first entry in the log buffer at 3:00:00 has a time stamp of 2:59:47, the next log file, when rolled, will have the following time stamp:

Log file 2

```
squid.log.mymachine.19980912.02h59m47s-19980912.06h00m00s.old
```

The beginning time stamp of a rolled log file is the time stamp of its first entry; the ending time stamp is the time of log rolling. The contents of a log file are always between the two time stamps. *Log files do not contain overlapping entries, even if successive time stamps appear to overlap.*

Rolling intervals

Log files are rolled at specific intervals relative to a given hour of the day. Two options control when log files are rolled:

- ✔ The rolling interval
- ✔ The offset hour, which is an hour between 0 (midnight) and 23

For example, if the rolling interval is 6 hours and the offset hour is 0 (midnight), then the logs will roll at midnight (00:00), 06:00, 12:00, and 18:00 each day. If the rolling interval is 12 hours and the offset hour is 3, then logs will roll at 03:00 and 15:00 each day.

Setting log file rolling options

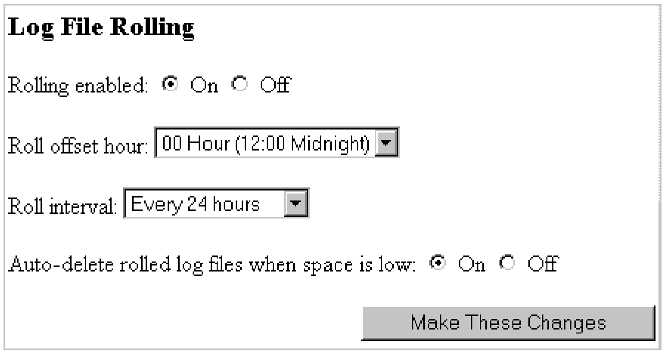
You can set log file rolling options by using the Traffic Manager UI or by editing a configuration file manually. Both procedures are provided below.

▼ To set log file rolling options from Traffic Manager:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Logging** button.

3 Scroll to the **Log File Rolling** section of the **Logging** page (shown below).



Log File Rolling

Rolling enabled: ● On ○ Off

Roll offset hour: 00 Hour (12:00 Midnight)

Roll interval: Every 24 hours

Auto-delete rolled log files when space is low: ● On ○ Off

4 Click the **Rolling enabled:On** button to turn on log file rotation.

5 In the **Roll offset hour** field, enter the time of day you want log file rolling to start.

You can enter any hour in the range 0 (midnight) to 23.

6 In the **Roll interval** field, enter the amount of time Traffic Server enters data in the log files before rotation takes place.

You can enter a value between 15 minutes and 24 hours.

7 Click the **Auto-delete rolled log files when space is low: On** button to enable auto deletion of rolled log files when available space in the log directory is low.

Auto deletion is triggered when the amount of free space available in the log directory is less than the headroom specified in the **Log Management** section of the **Logging** page.

8 Click the **Make These Changes** button.

▼ To set log file rolling options manually:

1 In a text editor, open the `records.config` file located in the `config` directory.

2 Edit the following variables:

records.config Variable	Description
proxy.config.log2.rolling_enabled	Set this variable to 1 to enable log file rotation.
proxy.config.log2.rolling_offset_hr	Set this variable to the time of day you want log file rolling to start.
proxy.config.log2.rolling_interval_sec	Set this variable to the rolling interval in seconds. The minimum value is 300 seconds (5 minutes).
proxy.config.log2.auto_delete_rolled_file	Set this variable to 1 to enable auto deletion of rolled files.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

Splitting event log files

By default, Traffic Server uses standard log formats and generates separate log files for HTTP/FTP, NNTP, and ICP transactions. Under most circumstances, this default behavior offers the most flexibility for collecting and analyzing log files. However, you can disable log splitting if you prefer to log all transactions for all protocols in the same log file.

Note If you are using a standard log file format (such as Squid or Netscape), Traffic Server always records HTTP and FTP transactions in the same log file. You cannot generate separate log files for transactions using these two protocols.

NNTP log splitting

When NNTP log splitting is enabled (the default behavior), Traffic Server records NNTP transactions in a separate log file with a name that contains `nntp`. For example, if you enable the Squid format, traffic Server records all NNTP entries in the `squid-nntp.log` file.

When you disable NNTP log splitting, Traffic Server records NNTP transactions in the same log file as HTTP and FTP transactions.

ICP log splitting

When ICP log splitting is enabled (the default behavior), Traffic Server records ICP transactions in a separate log file with a name that contains `icp`. For example, if you enable the Squid format, all ICP transactions are recorded in the `squid-icp.log` file.

When you disable ICP log splitting, Traffic Server records all ICP transactions in the same log file as HTTP and FTP transactions.

HTTP host log splitting

HTTP host log splitting enables you to record HTTP/FTP transactions for different origin servers in separate log files. When HTTP host log splitting is enabled, Traffic Server creates a separate log file for each origin server listed in the `log_hosts.config` file.

When NNTP, ICP, and host log splitting are all enabled, Traffic Server generates separate log files for HTTP/FTP transactions, based on the origin server, and places all NNTP and ICP transactions in their own respective log files.

For example, suppose the `log_hosts.config` file contains the two origin servers `uni.edu` and `company.com`, and the Squid format is enabled. Traffic Server generates the following log files:

Log file name	Description
`squid-uni.edu.log`	All HTTP and FTP transactions for `uni.edu`
`squid-company.com.log`	All HTTP and FTP transactions for `company.com`
`squid-nntp.log`	All NNTP transactions for all hosts
`squid-icp.log`	All ICP transactions for all hosts
`squid.log`	All HTTP and FTP transactions for other hosts

If you disable NNTP and ICP log splitting, NNTP and ICP transactions are placed in the same log file as HTTP and FTP transactions. Using the previous example hosts and assuming the Squid log format, Traffic Server generates these log files:

Log file name	Description
`squid-uni.edu.log`	All entries for `uni.edu`
`squid-company.com.log`	All entries for `company.com`
`squid.log`	All other entries

Traffic Server also enables you to create XML-based custom log formats that offer even greater control over log file generation based on protocol and hostname. For more information, see *Using custom formats, on page 161*.

Setting log splitting options

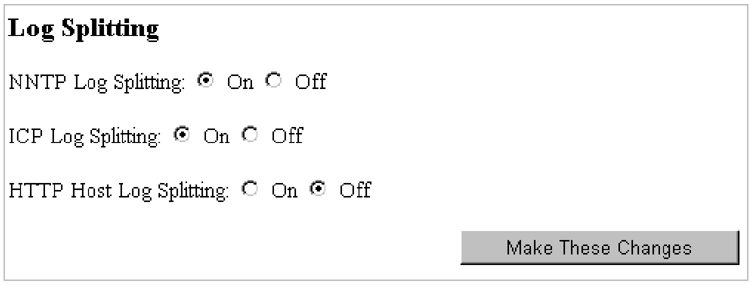
You can set log splitting options by using the Traffic Manager UI or by editing a configuration file manually. Both procedures are provided below.

▼ To set log splitting options from Traffic Manager:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Logging** button.

3 Scroll to the **Log Splitting** section of the **Logging** page (shown below).



Log Splitting

NNTP Log Splitting: ● On ○ Off

ICP Log Splitting: ● On ○ Off

HTTP Host Log Splitting: ○ On ● Off

4 Click the **NNTP Log Splitting: On** button to record all NNTP transactions in a separate log file.
Click the **NNTP Log Splitting: Off** button to record all NNTP transactions in the same log file as HTTP/FTP transactions.

5 Click the **ICP Log Splitting: On** button to record all ICP transactions in a separate log file.
Click the **ICP Log Splitting: Off** button to record all ICP transactions in the same log file as HTTP/FTP transactions.

6 Click the **HTTP Host Log Splitting: On** button to record all HTTP/FTP transactions for each origin server listed in `log_hosts.config` file in a separate log file.
Click the **HTTP Host Log Splitting: Off** button to record all HTTP/FTP transactions for each origin server listed in `log_hosts.config` file in the same log file.

7 Click the **Make These Changes** button.

▼ To set log splitting options manually:

1 In a text editor, open the `records.config` file located in the `config` directory.

2 Edit the following variables:

records.config Variable	Description
proxy.config.log2.separate_icp_logs	Set this variable to 1 to record all ICP transactions in a separate log file. Set this variable to 0 to record all ICP transactions in the same log file as HTTP/FTP transactions.
proxy.config.log2.separate_nntp_logs	Set this variable to 1 to record NNTP transactions in a separate log file. Set this variable to 0 to record all NNTP transactions in the same log file as HTTP/FTP transactions.
proxy.config.log2.separate_host_logs	Set this variable to 1 to record HTTP/FTP transactions for each host listed in log_hosts.config file in a separate log file. Set this variable to 0 to record all HTTP/FTP transactions for each host listed in the log_hosts.config file in the same log file.

3 Save and close the `records.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

Editing the log_hosts.config file

The default `log_hosts.config` file is located in Traffic Server's `config` directory. To record HTTP/FTP transactions for different origin servers in separate log files, you must specify each origin server's hostname on a separate line in the file.

Note If Traffic Server is clustered and if you enable log file collation, Inktomi recommends that you use the same `log_hosts.config` file on every Traffic Server node in the cluster.

▼ To edit the log_hosts.config file:

1 In a text editor, open the `log_hosts.config` file located in Traffic Server's `config` directory.

2 Enter the hostname of each origin server on a separate line in the file. For example:

```
webserver1
webserver2
webserver3
```

3 Save and close the `log_hosts.config` file.

4 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

5 Run the command `traffic_line -x` to apply the configuration changes.

Collating event log files

You can use Traffic Server's log file collation feature to keep all logged information in one place. This allows you to analyze Traffic Server as a whole rather than as individual nodes and to use a large disk that may only be located on one of the nodes in a cluster.

Traffic Server collates log files by using one or more nodes as log collation servers and all remaining nodes as log collation clients. When a Traffic Server node generates a buffer of event log entries, it determines whether it is the collation server or a collation client. The collation server node simply writes all log buffers to its local disk, just as it would if log collation were not enabled.

The collation client nodes prepare their log buffers for transfer across the network and send the buffers to the log collation server. When the log collation server receives a log buffer from a client, it writes it to its own log file as if it were generated locally. See *Figure 20*.

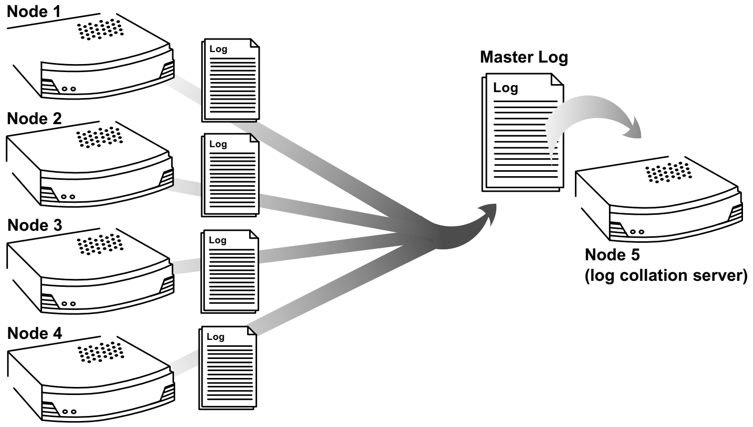


Figure 20 Log collation

If log clients cannot contact their log collation server, they write their log buffers to their local disks, into *orphan* log files. Orphan log files require manual collation. See *Figure 21*.

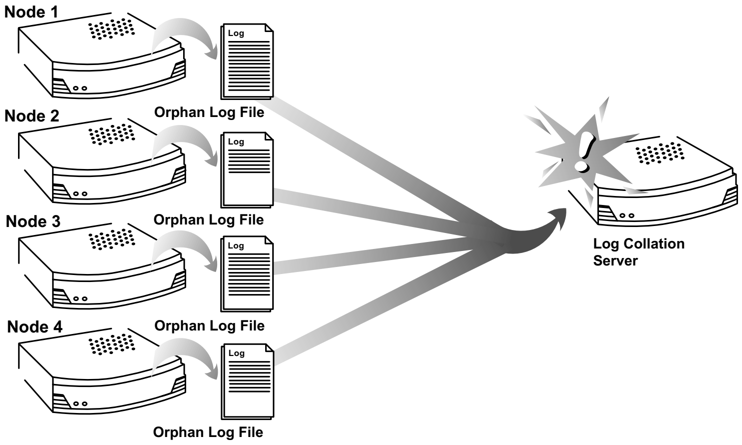


Figure 21 Orphan log files hold data that cannot be written to the log server

Log collation servers may be stand-alone or they may be part of a node running Traffic Server.

Note Log collation may have an impact on network performance. Because all nodes are forwarding their log data buffers to the single collation server, a bottleneck may occur in the network, where the amount of data being sent to a single node in the network exceeds the node's ability to process it quickly.

Collated log files contain time stamp information for each entry, but entries do not appear in the files in strict chronological order. You can sort collated log files before doing analysis.

Setting log collation options

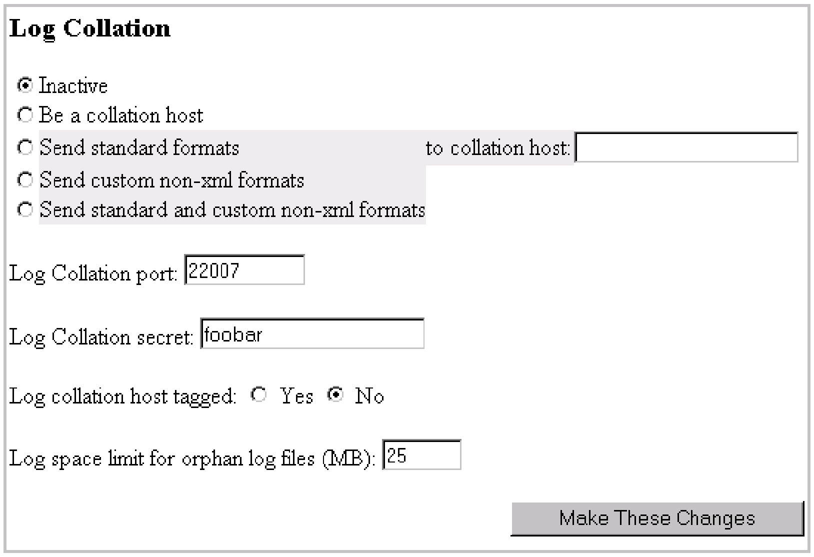
You can set log collation options by using the Traffic Manager UI or by editing a configuration file manually. Both procedures are provided below.

▼ To configure a Traffic Server node to be a collation server from Traffic Manager:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Logging** button.

3 Scroll to the **Log Collation** section of the **Logging** page (shown below).



Log Collation

● Inactive
○ Be a collation host
○ Send standard formats
○ Send custom non-xml formats
○ Send standard and custom non-xml formats

to collation host:

Log Collation port: 22007

Log Collation secret: foobar

Log collation host tagged: ○ Yes ● No

Log space limit for orphan log files (MB): 25

4 Click the **Be a collation host** button.

5 In the **Log Collation port** field, enter the port number that all nodes in a cluster must use to exchange event log entries. The default port number is 8085.

6 In the **Log Collation secret** field, enter the password used to validate logging data and prevent the exchange of arbitrary information.

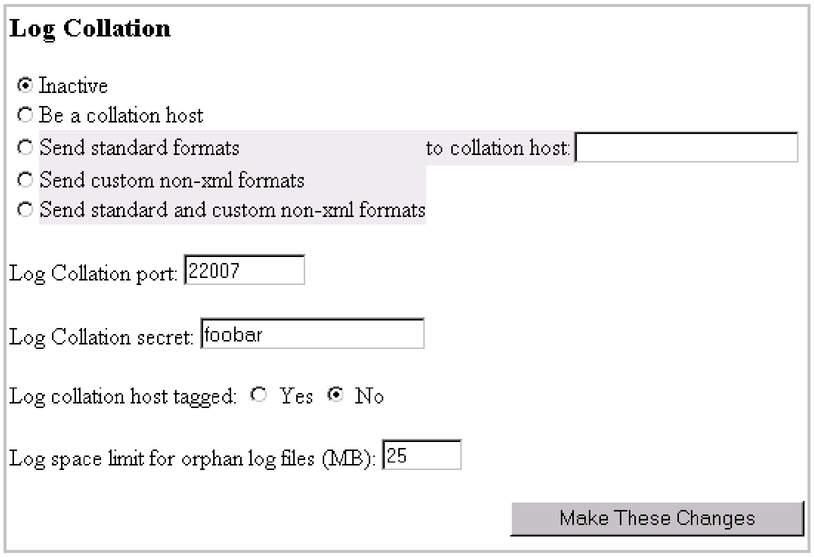
7 Click the **Make These Changes** button.

To configure a Traffic Server node to be a collation client from Traffic Manager:

1 From your browser, access the Traffic Manager UI (refer to *Accessing the Traffic Manager UI, on page 32*).

2 On the **Configure** tab, click the **Logging** button.

3 Scroll to the **Log Collation** section of the **Logging** page (shown below).



Log Collation

● Inactive
○ Be a collation host
○ Send standard formats
○ Send custom non-xml formats
○ Send standard and custom non-xml formats

to collation host:

Log Collation port: 22007

Log Collation secret: foobar

Log collation host tagged: ○ Yes ● No

Log space limit for orphan log files (MB): 25

4 Click one of the following buttons:

- **Send standard formats** to set the Traffic Server node as a collation client and send the active standard formats (such as Squid and Netscape) to the log collation server.

- **Send custom non-xml formats** to set the Traffic Server node as a collation client and send the custom *traditional* formats to the log collation server.

- **Send standard and custom non-xml formats** to set the Traffic Server node as a collation client and send both the standard formats (such as Squid and Netscape) *and* the custom *traditional* formats to the log collation server.

Note Traffic Server does not provide an option for custom XML-based specifications in the **Log Collation** section. XML-based specifications have their own way of specifying collation, refer to *Using XML-based custom formats, on page 162*.

5 In the **to collation host** field, enter the hostname of the collation server.
This could be the Traffic Server collation server or a standalone collation server.

6 In the **Log Collation port** field, enter the port number that all nodes in the cluster must use to exchange event log entries. The default port number is 8085.

7 In the **Log Collation secret** field, enter the password used to validate logging data and prevent the exchange of arbitrary information. This must be the same secret you set on the collation server.

8 In the **Log collation host tagged** field, select **Yes** if you want to preserve the origin of log entries in the collated log files.

9 In the **Log space limit for orphan log files** field, enter the maximum amount of space (in megabytes) you want to allocate to the logging directory on the collation client for storing orphan log files. (Orphan log files are created when the log collation server cannot be contacted).

10 Click the **Make These Changes** button.

▼ To set log collation options manually:

1 In a text editor, open the `records.config` file located in the `config` directory.

2 Edit the following variables:

records.config Variable	Description
proxy.config.log2.collation_host	Specify the collation server's hostname.
proxy.config.log2.collation_host_tagged	Set this variable to 1 if you want the hostname of the collation client that generated the log entry to be included in each entry. Set this variable to 0 if you do *not* want the hostname of the collation client that generated the log entry to be included in each entry.
proxy.config.log2.collation_port	Specify the port used for communication between the collation server and client.
proxy.config.log2.collation_secret	Specify the password used to validate logging data and prevent the exchange of arbitrary information.

3 Save and close the `records.config` file.

4 In a text editor, open the `lm.config` file located in the `config` directory.

5 Edit the following variables:

lm.config variable	Description
proxy.config.log2.collation_mode	Specify the option you want to use: ■ 0: Disable log file collation. ■ 1: Set host as a log collation server ■ 2: Set host as a log collation client and send log entries using standard formats to the collation server ■ 3: Set host as a log collation client and send custom non-XML formats to the collation server ■ 4: Set host as a log collation client and send both standard and custom non-XML formats to the collation server

6 Save and close the `lm.config` file.

7 In *UNIX*, make Traffic Server's `bin` directory your working directory.

In *Windows*, open a command prompt window and change to the `bin` directory (located in the Traffic Server installation directory).

8 Run the command `traffic_line -x` to apply the configuration changes.

Using a stand-alone collator

If you do not want the log collation server to be a Traffic Server node, you can install and configure a stand-alone collator (SAC) which can dedicate more of its power to collecting, processing, and writing log files.

Note The stand-alone collator is currently available for the UNIX platform only.

▼ To run a stand-alone collator:

1 Configure your Traffic Server nodes as log collation clients. Refer to *To configure a Traffic Server node to be a collation client from Traffic Manager:, on page 175*.

2 Copy the SAC binary from Traffic Server's `bin` directory to the machine serving as the stand-alone collator.

3 Copy the `records.config` file from a Traffic Server installation to a directory on the stand-alone collator.
The `records.config` file contains the log collation secret and port you specified when configuring Traffic Server nodes to be collation clients. The collation port and secret must be the same for all collation clients and hosts.

4 Enter the following command:

```
sac -c config_dir
```

where *`config_dir`* is the configuration directory in which you copied the `records.config` file on the stand-alone collator.

Recovering log files

If for any reason the Traffic Server logging system cannot write to your log collation server, it creates orphan log files on local disks, which you can collate manually into a large log file for the Traffic Server system.

▼ To move information from the orphan files into your central log files:

1 Copy the orphan files to the logging directory on your log collation server. (Examine local disks for the presence of any log files. These are the orphans.)

2 For each orphan file, type the following command at the command line:

```
logclean -m filename
```

This command collates records from the orphan file into your central log files.

3 Delete orphan log files from your local disks.

Note Collated log file entries do not appear in strict chronological order; they appear in the order received from the various nodes. Entries are time-stamped and you can sort them by time stamp.

Viewing logging statistics

Traffic Server generates the following statistics about the logging system that help you see:

- ✔ How many log files (formats) are currently being written.
- ✔ The current amount of space being used by the logging directory, which contains all of the event and error logs.
- ✔ The current number of events that have been written to log files. This counter represents one entry in one file. If multiple formats are being written, a single event will create multiple event log entries.
- ✔ The current number of events that have been written to the event error log.
- ✔ The approximate rate at which log file entries are being created.

You can view the statistics on the **Other** page from the **Monitor** tab in the Traffic Manager UI or retrieve them using the Traffic Line command-line interface. Refer to *Chapter 9, Monitoring Traffic*.

Appendix A

Traffic Manager Statistics

This appendix describes the statistics on the following Traffic Manager Monitor pages:

- *The Dashboard page, on page 180*
- *The Node page, on page 181*
- *The Graphs page, on page 182*
- *The Protocols page, on page 183*
- *The Cache page, on page 187*
- *The Other page, on page 188*
- *The MRTG page, on page 189*

The Dashboard page

The following table describes the statistics on the **Dashboard**.

Statistic/Field	Description
Node Name	The name of the Traffic Server node. If the node is in a cluster, the names of all the nodes in the cluster are displayed.
On/Off	The green On light Indicates that the traffic_server process is running (caching and proxying services are running). The green Off light indicates that the traffic_server process is *not* running.
Alarms	The green OK light indicates that no alarms are currently present on the node. The red light with an exclamation point indicates that alarms are present on the node. The yellow light with exclamation point indicates that there is a cluster problem.
Objects Served	The total number of objects served by the node.
Transactions per second	The number of transactions per second processed by the node.
	More Detail
Cache Hit Rate	The percentage of HTTP requests served from the cache, averaged over the past 10 seconds. This value is refreshed every 10 seconds.
Cache Hit Rate, Fresh	The percentage of HTTP requests for fresh objects in the cache, averaged over the past 10 seconds.
Cache Hit Rate, Refresh	The percentage of HTTP requests for expired objects that are revalidated, turn out to be still fresh, and served to clients; averaged over the past 10 seconds.
Errors	The percentage of HTTP requests that end in early hangups.
Aborts	The percentage of aborted HTTP requests.
Active Clients/servers	The current number of open HTTP client connections/open HTTP server connections.
Average Fresh Hit	The amount of time it takes Traffic Server to serve an HTTP request for an object that is fresh in the cache, averaged over 10 seconds.
Average Cached Miss	The amount of time it takes Traffic Server to serve an HTTP request for an object that is not in the cache, averaged over 10 seconds.
Client Throughput	The HTTP throughput, in MB per second.
Node IP Address	The IP address assigned to this node. If virtual IP addressing is enabled, several virtual IP addresses could be assigned to this node.
Virtual IP Address Mappings	If you have configured your Traffic Server system to use virtual IP addresses (refer to *Virtual IP failover, on page 84*), the virtual IP addresses for each node in a cluster display at the bottom of the More Detail page.

The Node page

The following table describes the statistics on the **Node** page.

Statistic	Description
	Cache
Document Hit Rate	The ratio of cache hits to total cache requests, averaged over 10 seconds. This value is refreshed every 10 seconds.
Bandwidth Savings	The ratio of bytes served from the cache to total requested bytes, averaged over 10 seconds. This value is refreshed every 10 seconds.
Cache Percent Free	The ratio of cache free space to total cache space.
	In Progress
Open Server Connections	The number of currently open origin server connections.
Open Client Connections	The number of currently open client connections.
Cache Transfers In Progress	The number of cache transfers (cache reads and writes) in progress.
	Network
Client Throughput (MB/Sec)	The number of MB per second passing through node (and cluster).
Transactions per second	The number of HTTP transactions per second.
	Name Resolution
Host Database Hit Rate	The ratio of host database hits to total host database lookups, averaged over 10 seconds. This value is refreshed every 10 seconds.
DNS Lookups Per Second	The number of DNS lookups per second.

The Graphs page

The **Graphs** page displays the same statistics displayed on the **Node** page (cache performance, current connections and transfers, network, and name resolution) in graphical format. In addition, you can use the **Graphs** page to display multiple statistics in one graph.

For information about using the **Graphs** button, refer to *The Graphs button, on page 113*. For a description of the statistics listed on the **Graphs** page, refer to *The Node page, on page 181*.

The Protocols page

The following table describes the statistics on the **Protocols** page.

Statistic	Description
	HTTP Transaction Frequency and Speeds
	Hits
Fresh	The percentage of hits that are fresh and their average transaction times.
Stale Revalidated	The percentage of hits that are stale and revalidated, turn out to be still fresh and served, and their average transaction times.
	Misses
Now Cached	The percentage of requests for documents that were not in the cache (but are now) and their average transaction times.
Server No Cache	The percentage of requests for documents that were not in the cache, but have server no-cache headers (cannot be cached); and their average transaction times.
Stale Reloaded	The percentage of misses that are revalidated, turn out to be changed, reloaded, and served; and their average transaction times.
Client no Cache	The percentage of misses with client no-cache headers and their average transaction times.
	Errors
Connection Failures	The percentage of connect errors and their average transaction times.
Other Errors	The percentage of other errors and their average transaction times.
	Aborted Transactions
Client Aborts	The percentage of client-aborted transactions, and their average transaction times.
Questionable Client Aborts	The percentage of transactions that could possibly be client aborted, and their average transaction times.
Partial Request Hangups	The percentage of early hangups (after partial requests) and their average transaction times.
Pre-Request Hangups	The percentage of pre-request hangups and their average transaction times.
Pre-Connect Hangups	The percentage of pre-connect hangups and their average transaction times.
	Other Transactions
Unclassified	The percentage of unclassified transactions and their average transaction times.
	HTTP Client
Total Document Bytes	The total amount of HTTP data served to clients since installation.
Total Header Bytes	The total amount of HTTP header data served to clients since installation.
Total Connections	The total number of HTTP client connections since installation.
Transactions In Progress	The total number of HTTP client transactions in progress.
	HTTP Server
Total Document Bytes	The total amount of HTTP data received from origin servers since installation.
Total Header Bytes	The total amount of HTTP header data received from origin servers since installation.
Total Connections	The total number of HTTP server connections since installation.
Transactions In Progress	The total number of HTTP server connections in progress.
	FTP
Open Connections	The number of open FTP connections.
PASV Connections Successes	The number of successful PASV connections since installation.
PASV Connections Failures	The number of PASV connection failures since installation.

Statistic	Description
PORT Connections Successes	The number of successful PORT connections since installation.
PORT Connections Failures	The number of PORT connection failures since installation.
	NNTP Client
Open Connections	The number of open NNTP connections.
Bytes Read	The number of NNTP client request bytes read since installation.
Bytes Written	The number of NNTP client bytes written since installation.
	NNTP Server
Open Connections	The number of currently open NNTP server connections.
Bytes Read	The number of bytes read from parent NNTP servers since installation.
Bytes Written	The number of NNTP bytes written to the cache since installation.
	Operations
Article Hits	The number of news article hits since installation.
Article Misses	The number of news article misses since installation.
Overview Hits	The number of overview hits since installation.
Overview Refreshes	The number of overview refreshes. An overview refresh occurs when the Traffic Server caches a group overview on demand (as opposed to an overview pull).
Group Hits	The number of group hits since installation.
Group Refreshes	The number of group refreshes since installation.
Posts	The number of article posts since installation.
Post Bytes	The amount of news data posted since installation.
Pull Bytes	The amount of article and overview data pulled since installation.
Feed Bytes	The amount of NNTP data received through news feeds since installation.
	ICP **Queries Originating From This Node**
Query Requests	The number of HTTP requests that generate ICP query messages.
Query Messages Sent	The total number of ICP query messages sent to ICP peers. This number is larger than the number of “ICP Query Requests” if there are multiple ICP peers.
Peer Hit Messages Received	The number of ICP peer hit messages received in response to ICP queries from this node.
Peer Miss Messages Received	The number of ICP peer miss messages received in response to ICP queries from this node.
Total Responses Received	The number of response messages received from ICP peers (siblings and parents).
Average ICP Message Response Time (ms)	The average time for an ICP peer to respond to an ICP query message from this node. This is a cumulative average value.
Average ICP Request Time (ms)	The average time for an HTTP request (that is sent to ICP) to receive an ICP response. This is a cumulative average value.
	Queries Originating From ICP Peers
Query Messages Received	The number of ICP query messages received from remote ICP peers (siblings and parents).
Remote Query Hits	The number of successful cache lookups in response to queries from ICP peers.
Remote Query Misses	The number of unsuccessful cache lookups in response to queries from ICP peers.
Successful Response Messages Sent to Peers	The number of successful ICP messages written in response to ICP queries from remote ICP peers.

Statistic	Description
WCCP 1.0 Router Statistics	
Router's IP address	The IP address of the WCCP router.
Router Status	Indicates if the WCCP router is running (UP) or not (DOWN).
WCCP 1.0 Node Statistics	
My IP address	The IP address of this Traffic Server node.
Percentage of traffic directed to this node	The percentage of port 80 traffic directed toward this Traffic Server node.
Number of heartbeats received	The WCCP router sends a heartbeat message to this node every 10 seconds. A heartbeat message is a short message that indicates to the recipient, “I am up and running.” This value is the cumulative number of heartbeats received from the router.
WCCP 1.0 Protocol Statistics	
Enabled	Indicates if WCCP is enabled on this node.
Leader's IP address	The IP address of the leader node.
Number of active nodes	The number of active nodes in the WCCP cache farm.
WCCP 2.0 Configuration Information	
Security Enabled	Indicates if WCCP security is enabled.
Multicast Enabled	Indicates if multicast mode is enabled.
Multicast Address	The IP multicast address (if multicast mode is enabled).
Multicast TTL	The multicast Time To Live value specifies how many hops the router is from the cache.
WCCP 2.0 Services Information	
HTTP	Indicates if transparent redirection of HTTP traffic is enabled.
NNTP	Indicates if transparent redirection of NNTP traffic is enabled.
RTSP	Indicates if transparent redirection of RTSP traffic is enabled (Media-IXT option).
PNA	Indicates if transparent redirection of PNA traffic is enabled (Media-IXT option).
WCCP 2.0 HTTP Statistics	
My IP	The IP address of the Traffic Server receiving HTTP traffic from a WCCP router.
Number of buckets assigned to this node	The number of buckets assigned to this Traffic Server node by the WCCP router.
Service Id	The number that identifies the protocol in the router command. 0 is used for HTTP.
Leader's IP	The IP address of the leader node in the WCCP cache farm.
Number of Caches	The number of nodes in the WCCP cache farm.
Number of Routers	The number of WCCP routers sending traffic to the Traffic Server.
Router[0] IP address Router[1] IP address	The IP address of the router sending traffic to the Traffic Server. If there is more than one router sending traffic, the IP addresses of all other routers are also listed.
Router[0] Received ID Router[1] Received ID	The received ID of the router sending traffic to the Traffic server. If there is more than one router sending traffic, the received IDs of all other routers are also listed.

Statistic	Description
	WCCP 2.0 NNTP Statistics
My IP	The IP address of the Traffic Server receiving NNTP traffic from a WCCP router.
Number of buckets assigned to me	The number of buckets assigned to the Traffic Server by the WCCP router.
Service Id	The number that identifies the protocol in the router command. 1 is used for NNTP.
Leader's IP	The IP address of the leader node in the WCCP cache farm.
Number of Caches	The number of nodes in the WCCP cache farm.
Number of Routers	The number of WCCP routers sending traffic to the Traffic Server.
Router[0] IP address Router[1] IP address	The IP address of the router sending traffic to the Traffic Server. If there is more than one router sending traffic, the IP addresses of all other routers are also listed.
Router[0] Received ID Router[1] Received ID	The received ID of the router sending traffic to the Traffic server. If there is more than one router sending traffic, the received IDs of all other routers are also listed.

The Cache page

The following table describes the statistics on the **Cache** page.

Statistic	Description
Bytes Used	The number of bytes currently used by the cache.
Cache Size	The number of bytes allocated to the cache.
	RAMCACHE
Bytes	The total size of the RAM cache in bytes.
Hits	The number of document hits from the RAM cache.
Misses	The number of document hits from the cache disk.
	LOOKUPS
In Progress	The number of cache lookups for ICP hits in progress.
Hits	The number of completed cache lookups (for ICP hits) since installation.
Misses	The number of ICP misses since installation.
	READS
In Progress	The number of cache reads in progress (NNTP, HTTP, and FTP).
Hits	The number of cache reads completed since installation (NNTP, HTTP, and FTP).
Misses	The number of cache read misses since installation (NNTP, HTTP, and FTP).
	WRITES
In Progress	The number of cache writes in progress (NNTP, HTTP, and FTP).
Successes	The number of successful cache writes since installation (NNTP, HTTP, and FTP).
Failures	The number of cache write failures since installation (NNTP, HTTP, and FTP).
	UPDATES
In Progress	The number of HTTP document updates in progress. An update occurs when the Traffic Server revalidates a document, finds it to be fresh, and updates the document header.
Successes	The number of successful cache HTTP updates completed since installation.
Failures	The number of cache HTTP update failures since installation.
	REMOVES
In Progress	The number of document removes in progress. A remove occurs when the Traffic Server revalidates a document, finds it to be deleted on the origin server, and deletes it from the cache (includes NNTP, HTTP, and FTP removes).
Successes	The number of successful cache removes completed since installation (includes NNTP, HTTP, and FTP removes).
Failures	The number of cache remove failures since installation (includes NNTP, HTTP, and FTP removes).

The Other page

The following table describes the statistics on the **Other** page.

Statistic	Description
Host DataBase	
Total Lookups	The total number of lookups in the Traffic Server host database since installation.
Total Hits	The total number of host database lookup hits since installation.
Average TTL (min)	The average time to live in minutes.
DNS	
Total Lookups	The total number of DNS lookups (queries to name servers) since installation.
Successes	The total number of successful DNS lookup since installation.
Average Lookup Time (ms)	The average DNS lookup time.
Cluster	
Bytes Read	The number of bytes read by this node from other nodes in the cluster since installation.
Bytes Written	The number of bytes this node has written to other cluster nodes since installation.
Connections Open	The total number of intracluster connections opened since installation.
Total Operations	The total number of cluster transactions since installation.
Network Backups	The number of times this node encountered intracluster network congestion and reverted to proxy-only mode since installation.
Clustering Nodes	The number of clustering nodes.
SOCKS	
Unsuccessful Connections	The number of unsuccessful connections to the SOCKS server since installation.
Successful Connections	The number of successful connections to the SOCKS server since installation.
Connections in progress	The number of SOCKS connections in progress.
Logging	
Currently Open Log Files	The number of event log files (formats) that are currently being written.
Space Used For Log Files	The current amount of space being used by the logging directory, which contains all of the event and error logs.
Number of Access Events Logged	The current number of access events that have been written to log files. This counter represents one entry in one file, so that if multiple formats are being written, a single access will create multiple event log entries.
Number of Access Events Skipped	The number of skipped access events.
Number of Error Events Logged	The current number of events that have been written to the event error log.

The MRTG page

The MRTG graphs are described in an online document. To view descriptions about the various graphs available, click the **more info** link at the bottom of the **MRTG** page.

Appendix B

Traffic Manager Configuration Options

This appendix describes the configuration options in the following Traffic Manager Configure pages:

The Server Basics page

The following table describes the configuration options on the **Server Basics** page.

Option	Description
	Traffic Server
Traffic Server on/off	Turns the `traffic_server` process on or off, which shuts down all caching and proxying services on a node-by-node basis. You can turn the `traffic_server` process on or off only one node at a time. Normally, the `traffic_server` process should remain on. However, you must turn the Traffic Server off before performing certain maintenance tasks.
Traffic Server Name	Displays the hostname of your Traffic Server or the hostnames of all the nodes in a cluster.
Traffic Server Port	Specifies the port number by which all browsers can connect to the proxy process that runs on the Traffic Server system. The port must be dedicated. Note: The default proxy port number is 8080. The Traffic Server's manager process must run as root to bind to port numbers less than 1024.
Traffic Server User ID	Specifies the user ID for the Traffic Server's proxy process. You cannot edit your user ID on this page. Note: Windows should always be run as a LocalSystem account.
Local Domain Expansion	Turns local domain expansion on so that Traffic Server can attempt to resolve unqualified hostnames by expanding to the local domain. For example, if a client makes a request to an unqualified host named `host_x`, and if the Traffic Server's local domain is `y.com`, the Traffic Server will expand the hostname to `host_x.y.com`.
.com Domain Expansion	Turns `.com` domain expansion on so that Traffic Server can attempt to resolve unqualified hostnames by redirecting them to the expanded address, prepended with `www.` and appended with `.com`. For example, if a client makes a request to `inktomi`, the Traffic Server redirects the request to `www.inktomi.com`. Note: If local domain expansion is set to on, the Traffic Server attempts local domain expansion before `.com` domain expansion; Traffic Server tries `.com` domain expansion only if local domain expansion fails.
	Web Management
Traffic Manager Restart	Restarts the traffic_manager process, which restarts the entire Traffic Server cluster. You must restart this process to effect changes you have made to port numbers and virtual IP addresses. Restart takes about 15 seconds and caching and proxying services are disabled during this time.
Traffic Manager Port	Specifies the port number by which the administrator's browser can connect to the traffic_manager process. The port must be on the Traffic Server and it must be dedicated to Traffic Server use. The default port number is 8081. Note: The traffic_manager process must run as root to bind to port numbers less than 1024.
Refresh rate in Monitor mode	Select a refresh rate for the statistics displayed on Traffic Manager's **Monitor** tab.

Virtual IP Addressing	
Virtual IP on/off	Set virtual IP addressing on or off. **Caution**: If virtual IP addressing is disabled, Traffic Server nodes cannot cover each other's failures.
Edit virtual IP addresses	Click this link to edit your list of virtual IP addresses (first assigned when you installed the Traffic Server). Changes are not effective until you restart the traffic_manager process (click **Restart** on the same page). **Caution**: Incorrect IP addressing can disable your system. Make sure you understand how virtual IP addresses work before changing them.
Auto-Configuration of Browsers	
Auto-configuration file	Web browsers use the Traffic Server by specifying a preference to use a proxy server, usually through an auto-configuration file. Click this link to create or edit a script file that contains information for automatically configuring client browsers to connect to the Traffic Server. For information on setting your browser to use a proxy, such as the Traffic Server, see your browser documentation. If you are using the transparency option, auto-configuration files are not needed.
Auto-configuration port	Specify the port to use for downloading the auto-configuration file. The port cannot be assigned to any other process. Changes are not effective until you restart the traffic_manager process (click Restart on the same page).
Throttling of Network Connections	
Maximum Number of Connections	Specifies the maximum number of network connections accepted by the Traffic Server. Setting a throttling limit on the Traffic Server helps to prevent system overload when traffic bottlenecks develop. When network connections reach this maximum, new connections are queued until existing connections close. Note: This option is not supported on Traffic Server for Windows.
Configure Load Shedding	
Maximum Concurrent Client Connections	Specifies the maximum number of client connections accepted by a Traffic Server running in transparent proxy caching mode. Note: This option displays only when transparent proxy caching is enabled.
SNMP	
SNMP Agent On/Off	Select On to enable MIB access on your Traffic Server or Traffic Server cluster. Traffic Server's SNMP agent supports access to two management information bases (MIBs): MIB-2 (a standard MIB) and the Inktomi Traffic Server MIB. Descriptions of the Traffic Server MIB variables are provided in the `inktomi-ts-mib.my` file in Traffic Server's `config/mibs` directory (UNIX) and the `config\mibs` directory (Windows). The Traffic Server MIB contains both node-specific and cluster-wide information. You should configure your system so that only certain hosts can access these MIBs. Configure access control and SNMP trap destinations in the `snmpd.cnf` file in Traffic Server's `config` directory. See *snmpd.cnf, on page 291*.

Customizable Response Pages	
Traffic Server should suppress generated response pages:	If Traffic Server detects an HTTP problem with a particular client transaction (such as, unavailable origin servers, authentication requirements, and protocol errors), it sends an HTML response page to the client browser. Traffic Server has a set of hard-coded default response pages that explain each HTTP error in detail to the client. You can suppress Traffic Server's response pages (this might be particularly useful when Traffic Server is running transparently and you do not want to indicate the presence of a cache). Select **Never** to always send response pages. Select **Always** to always suppress response pages. When response pages are suppressed, clients see `Document contains no data` dialog boxes on their screens, instead of Traffic Server response pages. Select **When Transparent** to suppress response pages only if transparency is enabled.
Control customizable response pages:	Select **Turn Off Customizable Response Pages** to send the default response pages to clients. Select **Enable Default Custom Response Pages** to send customized response pages to your clients. You can customize the error message text and format to provide a different look and feel or to explain the errors in a different language. Select **Enable Language-Targeted Custom Response Pages** to send the customized response pages to clients in the language specified in the Accept-Language header. Select **Log Customization Activity to Error Log** to send a message to the error log each time a customized response page is used or modified.
Control Response Page Template Directory	Specifies the directory where customized response page templates reside. The default is Traffic Server's `config/body_factory` (UNIX) directory or `config\body_factory` (Windows).

The Protocols page

The following table describes the configuration options on the **Protocols** page.

Option	Definition
	HTTP
Keep-Alive Timeout Inbound	Specifies how long the Traffic Server should keep connections to clients open for a subsequent request after a transaction ends. Each time the Traffic Server opens a connection to accept a client request, it handles the request, then keeps the connection alive for the timeout period you specify. If the client does not make another request before the timeout expires, the Traffic Server closes the connection. If the client does make another request, the timeout period starts over. Note: The client can close the connection at any time.
Keep-Alive Timeout Outbound	Specifies how long the Traffic Server should keep connections to origin servers open for a subsequent transfer of data after a transaction ends. Each time the Traffic Server opens a connection to download data from an origin server, it downloads the data, then keeps the connection alive for the timeout period you specify. If the Traffic Server does not need to make a subsequent request for data before the timeout expires, it closes the connection. If it does, the timeout period starts over. Note: The origin server can close the connection at any time.
Inactivity Timeout Inbound	Specifies how long the Traffic Server should keep connections to clients open if a transaction stalls. If the Traffic Server stops receiving data from a client or the client stops reading the data, the Traffic Server closes the connection when this timeout expires. Note: The client can close the connection at any time.
Inactivity Timeout Outbound	Specifies how long the Traffic Server should keep connections to origin servers open if the transaction stalls. If the Traffic Server stops receiving data from an origin server, it will not close the connection until this timeout has expired. Note: The origin server can close the connection at any time.
Activity timeout (inbound)	Specifies the maximum time the Traffic Server should remain connected to a client. If the client does not finish making a request—reading and writing data—before this timeout expires, the Traffic Server closes the connection. Note: The client can close the connection at any time.
Activity Timeout Outbound	Specifies the maximum time the Traffic Server should wait for fulfillment of a connection request to an origin server. If the Traffic Server does not establish connection to an origin server before this timeout expires, the Traffic Server terminates the connection request. Note: The origin server can close the connection at any time.
Remove common headers	You can remove any of these headers (which accompany transactions) to protect the privacy of your site: The **From** header (identifies the client's e-mail address) The **Referer** header (identifies the Web link followed by the client) The **User-Agent** header (identifies the agent—usually a browser—making the request) The **Cookie** header (the cookie field is often used to identify the user that made the request)
Comma-separated list of other headers to remove	Enter a comma-separated list of headers (other than from, referer, user-agent, and cookie) that you want to remove.

Option	Definition
Insert Client-ip	Select **Insert Client-ip** to retain client IP addresses from headers.
Remove Client-ip	Select **Remove Client-ip** to remove client IP addresses from headers for more privacy.
	NNTP
NNTP Server on/off	Enables the Traffic Server to cache and serve news articles. **Note**: After turning NNTP on or off, you must restart the Traffic Server cluster to effect the change.
NNTP Server Port	Specifies the port that the Traffic Server uses for serving NNTP requests. The default port is 119. Note: The traffic_manager process must run as root to connect to port numbers less than 1024.
Connect Message (posting allowed)	The message that is displayed to news readers when they connect to the Traffic Server, if posting is allowed.
Connect Message (posting not allowed)	The message that is displayed to news readers when they connect to the Traffic Server, if posting is *not* allowed.
NNTP option: Posting	Allows users to post NNTP articles to parent NNTP servers.
NNTP option: Access Control	Turns access control on or off. You can refine access control by configuring the `nntp_access.config` file in Traffic Server's `config` directory; see *nntp_access.config, on page 251*. If you are using an authentication server, you must enter its name and port (see *Authentication Server Host, on page 197*).
NNTP option: NNTP V2 Authentication Server	Supports NNTP version 2 authentication. Use this option only if you are certain that all of your client authentication supports version 2.
NNTP option: Run Local Authentication Server	Runs an authentication program on this Traffic Server node. You can configure which clients must be authenticated in the `nntp_access.config` file. See *page 251*.
NNTP option: Clustering	Allows cluster-wide NNTP caching.
NNTP option: Allow Feeds	Allows the Traffic Server to accept feeds of news articles from feed or push groups. **Caution**: If Traffic Server is clustered, configure your news server to send feeds to only *one* node in the cluster. Feed and push groups are designated in the `nntp_servers.config` file. The Traffic Server does not cache news articles from feed groups; instead, it receives feeds of news articles as the parent NNTP server receives feeds. Push groups are groups for which the Traffic Server can both retrieve articles on demand and receive news feeds. See *nntp_servers.config, on page 252* for information about designating news groups as push or feed in the `nntp_servers.config` file.
NNTP option: Access Logs	Configures Traffic Server to log NNTP transactions in its event logs.
NNTP option: Background Posting	Configures Traffic Server to post NNTP articles to parent NNTP servers in the background.
NNTP option: Obey Cancel Control Messages	Sets the Traffic Server to obey `cancel` control messages. When the Traffic Server gets a cancel control message, it deletes the corresponding article from the cache. You do not need to enable this option if the Traffic Server is caching articles on demand (if there are no feed groups). For all non-feed news groups, the Traffic Server actively polls parent NNTP servers for cancelled articles. See the **Check for Cancelled Articles** option, below.

Option	Definition
NNTP option: Obey Newgroups Control Messages	Configures Traffic Server to obey `newgroup` control messages. Note: Traffic Server actively polls parent NNTP servers for new groups; see the Check for New Groups option, below.
NNTP option: Obey Rmgroups Control Messages	Sets the Traffic Server to obey `rmgroup` (remove group) control messages.
Inactivity Timeout	Sets the number of seconds that idle connections can remain open. This timeout should be at least 3 minutes.
Check for New Groups Every	The Traffic Server regularly polls parent NNTP servers for new groups. This option is the time period between checks. The parent group lists change slowly, so you do not need to check them frequently. Note: You must list the hosts you want to poll in the `nntp_servers.config` file.
Check for Cancelled Articles Every	The Traffic Server regularly polls all non-feed news groups on the parent NNTP servers for cancelled articles. This option is the amount of time between checks. Checking for new articles should not be done too frequently as it involves communication with the parent NNTP server.
Check Parent NNTP Server Every	The Traffic Server regularly polls the parent NNTP server for new articles. This is the amount of time between polls.
Check Cluster Every	Specifies how often the Traffic Server polls other Traffic Server nodes in the cluster to see if new articles have appeared.
Check Pull Groups Every	Sets how often the Traffic Server checks pull groups for new articles (and "pulls" the new articles if they exist). The Traffic Server actively pulls (caches) news articles from pull groups, rather than waiting for user requests. Pull groups are designated in the `nntp_servers.config` file.
Authentication Server Host	Specifies the name of the host machine running the authentication server. If the host machine is the Traffic Server, enter localhost.
Authentication Server Port	The locally run authentication server accepts connections on this port. If the authentication server is remote, the Traffic Server connects to it on this port.
Local Authentication Server Timeout	The authentication server will abort an authorization operation if it does not complete in this amount of time. The client can retry the operation. Refer to *nntp_access.config, on page 251* for information about configuring authentication servers.
Client Speed Throttle	Clients are limited to downloading no more than this number of bytes per second. A throttle of `0` (zero) means that downloading is not limited.
	HTTPS
Restrict SSL connections to ports	Configures Traffic server to restrict SSL connections to certain ports, thereby containing attacks to designated ports

Option	Definition
	FTP
FTP connection mode	An FTP transfer requires two connections: a control connection to inform the FTP server of a request for data and a data connection to send the data. The Traffic Server always initiates the control connection. FTP mode determines whether the Traffic Server or the FTP server initiates the data connection. **PASV/PORT** = The Traffic Server attempts to use PASV connection mode. If PASV mode fails, the Traffic Server then tries PORT mode, initiates the data connection, and then the FTP server accepts it. **PASV only** = The Traffic Server initiates the data connection to the FTP server and the FTP server accepts it. This mode is firewall friendly, but some FTP servers do not support it. **PORT only** = The FTP server initiates the data connection and the Traffic Server accepts it. The default value is PASV/PORT.
FTP inactivity timeout (seconds)	Specifies how long the Traffic Server waits for a response from the FTP server. If the FTP server does not respond within the amount of time you specify, the Traffic Server abandons the client's request for data.
Anonymous FTP password	Specifies an anonymous password for FTP servers that require a password for access.

The Cache page

The following table describes the configuration options on the **Cache** page.

Option	Description
	Cache Activation
Enable HTTP caching	Configures the Traffic Server to cache objects retrieved via HTTP.
Enable FTP caching	Configures the Traffic Server to cache FTP objects retrieved via HTTP.
Enable NNTP caching	Configures the Traffic Server to cache objects retrieved via NNTP
Ignore user requests to bypass cache	If clients stipulate that their requests are not to be served from the cache, ignore the stipulation (ignore client Cache Control: `no-cache` headers).
	Storage
Maximum HTTP/FTP object size in bytes	Specifies the maximum size of HTTP or FTP objects to be cached. **Caution**: If you enter `0` (zero), there is no limit on the maximum HTTP or FTP object size.
Maximum number of alternate versions (HTTP)	Specifies the maximum number of HTTP alternates that Traffic Server can cache. **Caution**: If you enter `0` (zero), there is no limit on the number of alternates cached. If a popular URL has thousands of alternates, you might observe increased cache hit latencies (transaction times) as Traffic Server searches over the thousands of alternates for each request. In particular, some URLs can have large numbers of alternates due to cookies. If Traffic Server is set to vary on cookies, you might encounter this problem.
View cache storage configuration	Click this link to see a list of the files or hard disk partitions allotted to cache storage and their sizes. Note: Raw partitions may not have an associated size.
	Freshness
Verify freshness by checking	Configures the Traffic Server to ask the origin server to verify the freshness of objects before serving them. Configures the Traffic Server to ask for verification: When the object has expired When the object has expired or if the object has no expiration date Always Never
Minimum freshness information for a document to be cacheable	Specifies the minimum freshness information required to consider a document cacheable: An explicit lifetime A last-modified time Nothing
If an object has no expiration date	Some origin servers do not stamp the objects they serve with an expiration date, but you can control how long these objects remain in the cache by specifying: The minimum time they can remain in the cache (from 15 minutes to two weeks). The maximum time they can remain in the cache (from 15 minutes to two weeks).
FTP cached objects expire	FTP objects carry no time stamp or date information and stay in the Traffic Server cache pending removal on a schedule you specify (from 15 minutes to two weeks).

Option	Description
Internet Explorer requests force a check with the origin server	Certain versions of Microsoft Internet Explorer do not request cache reloads from reverse proxies and transparent caches when the user presses the browser **Refresh** button. This can prevent content from being loaded directly from the origin servers. You can configure Traffic Server to treat Microsoft Internet Explorer requests more conservatively, providing fresher content at the cost of serving fewer documents from cache. The following options are available: **never** = never force a freshness check with the origin server **for IMS revalidation requests** = only force a freshness check for IMS (If Modified Since) revalidation requests **always** = always force a freshness check with the origin server
	Variable Content
Do not cache	Some origin servers answer requests to the same URL with a variety of objects. The content of these objects can vary widely, according to whether a server delivers content for different languages, targets different browsers with different presentation styles, or delivers variable content at different times of the day. You can set the Traffic Server to refuse to cache objects served in response to URLs that contain a question mark, a semi-colon, cgi, or end in .asp.
Enable Alternates	Configures Traffic Server to cache alternate versions of HTTP documents.
Vary on these HTTP header fields:	Using document header information, Traffic Server can compare cached document specifications against requested specifications to determine if the correct alternate version of the document is in the cache. For example, a document header can specify that the document targets a specific browser, but any browser can request the document from Traffic Server. If a requested document's fields do not match a cached document's fields, the Traffic Server does not serve the document from its cache, and instead retrieves a fresh copy from the origin server. If you select the **Enable Alternates** option, you can specify values to match for the following fields: If the request is for text = The default value is `user-agent` and `cookie` Note: Some documents can have thousands of alternate cookie versions. If you choose to vary on cookies, Inktomi recommends that you limit the number of alternates cached. See *Storage, on page 199*. If the request is for images = Images are rarely personalized. If the request is for anything other than text or images.
Cache responses to requests containing Cookies for:	You can configure Traffic Server to cache responses to requests that contain cookies for: No content-types All content-types Only image-content types Content-types which are not text

The Security page

The following table describes the configuration options on the **Security** page.

Option	Description
Control Access to the Traffic Server Manager	
Authentication (basic) on/off	Sets basic authentication on or off. When on, Traffic Server checks the administrator ID and password or user name and password (if administrator accounts have been configured) whenever a user logs on to the Traffic Manager UI.
Administrator's ID	Specifies the administrator login ID. (The ID is not checked if you turn basic authentication off.) The administrator has access to both Configure and Monitor modes in the Traffic Manager UI. The administrator ID cannot contain a colon (:).
Change Administrator's Password	Click this link to change the administrator password. (The password is not checked if you turn basic authentication off). During installation, you select an administrator password. The installer automatically encrypts the passwords and stores the encryptions in the `records.config` file so that no one can read them. Each time you change passwords in the Traffic Manager UI, Traffic Serve encrypts the passwords and stores the encryptions in the `records.config` file. **Note**: If you forget the administrator password and cannot access the Traffic Manager UI Configure pages, set the administrator password variable (`proxy.config.admin.admin_password`) to NULL in the `records.config` file, and then enter a new password in this field. Setting the password variable to NULL in the configuration file means that a password is not needed to access the Traffic Manager UI. You cannot set passwords in the `records.config` file because the password variables can only contain password encryptions or NULL.
Additional Users	Click this link to create a list of administrator accounts that defines who has access to the Traffic Manager UI and which activities they can perform. You can disable Traffic Manager access for specific users, allow specific users to view statistics only, view statistics and view configuration options, or view statistics and *change* configuration options. Refer to *Creating a list of administrator accounts, on page 138* for information about creating administrator accounts.
SSL On / Off	This button displays only if you have obtained and installed an SSL certificate to use for Traffic Manager connections. Click On to enable the SSL option to provide protection for remote administrative monitoring and configuration.
Firewall Configuration	
SOCKS on/off	If you have no firewall or if the Traffic Server is outside your firewall, set SOCKS off (the default value). If your Traffic Server is inside the firewall, set SOCKS on.
SOCKS server IP address	Specifies the IP address of your SOCKS server.
SOCKS server port	Specifies a port by which the Traffic Server can connect to your SOCKS server. The port must be dedicated. Note: The traffic_manager process must run as root to connect to port numbers less than 1024.

Option	Description
SOCKS timeout (seconds)	Specifies how long (in seconds) the Traffic Server must wait for the SOCKS server to respond. If the SOCKS server does not respond within the amount of time specified, the Traffic Server drops the connection.
SOCKS List	Displays a page where you can specify the origin servers to which you want to connect without going through the SOCKS server. The list of origin servers is recorded in the `socks.config` file. If the SOCKS option is off, the Traffic Server does not read this file. In the **IP Range** field, you can enter a single IP address, a range of IP addresses, or a combination of IP addresses and ranges separated by commas. Note: Any machine not identified in the SOCKS configuration file is considered to be outside the firewall and the Traffic Server will connect to it only by going through the SOCKS server.

The Routing page

The following table describes the configuration options on the **Routing** page.

Option	Description
	Parent Caching
Parent Caching on/off	Sets HTTP parent caching on so that Traffic Server can participate in an HTTP cache hierarchy. You can point your Traffic Server at a parent network cache (either another Traffic Server or a different caching product) to form a cache hierarchy where a child cache relies upon a parent cache in fulfilling client requests.
Parent Cache	Specifies the identify a parent cache and parent cache port using the following format: `parent_name:port_number`. The port must be dedicated. If the Traffic Server cannot find a requested object in its own cache, it searches the parent cache before searching the internet. If you want parent failover, you can specify more than one parent cache: `parent1:port1; parent2:port2` Note: the `traffic_manager` process must run as "root" to bind to port numbers less than 1024.
	ICP
ICP Mode	The following options enable or disable ICP mode: Only Receive Queries Send/Receive Queries Disabled
ICP Port	Specifies the port that you want to use for ICP messages. The default is 3130.
ICP Multicast enabled on/off	If your Traffic Server has a multicast channel connection to its ICP peers, it can send ICP messages through multicast if you enable this option.
ICP Query Timeout	Specifies the timeout for ICP queries in seconds.
ICP Peers	Click this link to view or modify the Traffic Server's ICP hierarchy. For ICP to work, the Traffic Server must recognize its ICP neighbors (siblings and parents). Refer to *Identifying ICP Peers, on page 98*.
	Reverse proxy
Reverse Proxy	Enables or disables HTTP reverse proxy caching mode. For FTP reverse caching mode, refer to *FTP Reverse Proxy, on page 73*. If you enable reverse proxy, Traffic Server is a reverse proxy for the origin servers specified in mapping rules (in the `remap.config` file).
Retain Client Host Header	Select Yes to retain the client host header in a request (Traffic Server will not translate the client host header).
	Mapping/Redirection
Serve Mapped Hosts Only	If you select Yes, Traffic Server does not serve requests from the cache to unspecified origin servers (origin servers not listed in the `remap.config` file. See *Understanding reverse proxy caching, on page 66*. If you select No, Traffic Server serves requests from unspecified origin servers as a normal proxy cache.

Option	Description
Edit Mapping Rules	Click this link to view, modify, or add mapping rules. See *Understanding reverse proxy caching, on page 66* for information on mapping rules.
Redirect requests without Host header to URL	Specifies an alternate URL to which to direct incoming requests from older clients that do not provide a `Host:` header. The best solution is to set this option to a page that explains the situation to the user and advises a browser upgrade or provides a link directly to the origin server, bypassing the Traffic Server. Alternatively, you can specify a map rule that maps requests without `Host:` headers to a particular server.

The Host Database page

The following table describes the configuration options on the **Host Database** page.

Option	Description
	Host Database Management
Lookup timeout	Specifies the DNS lookup timeout in seconds. You can enter 5, 10, 15, 20, or 30 seconds.
Foreground timeout	Specifies how long DNS entries can remain in the database before they are flagged as stale. For example, if this timeout is 24 hours, and a client requests an entry that has been in the database for 24 hours or longer, the Traffic Server will refresh the entry before serving it. You can set the background timeout (see next item) to refresh entries in the background, before objects become stale. **Caution**: Setting the foreground timeout too low might slow response time. Setting it too high risks accumulation of incorrect information. Setting the foreground timeout to greater than or equal to the background timeout disables background refresh.
Background timeout	Specifies how long DNS entries can remain in the database before they are flagged as entries to refresh in the background. These entries are still fresh, so they can be refreshed after they are served, rather than before. Example: The foreground refresh timeout interval is 24 hours and the background timeout is 12 hours. A client requests an object from `my.com` and 16 hours later a client makes a second request for an object from `my.com`. The DNS entry for `my.com` has not been refreshed in the foreground because the entry is not yet 24 hours old. But since the background timeout has expired, the Traffic Server will first serve the client's request, then refresh the entry in the background.
Invalid host timeout	Specifies how long the proxy software should remember that a hostname is invalid. This is often called negative DNS caching. For example, if a client specifies an invalid hostname, the Traffic Server informs the client that it could not resolve the hostname and the Traffic Server gets another request for the same hostname. If the Traffic Server still remembers the bad hostname, it will not try to look it up again, but will simply send another `invalid host name` message to the client.
Re-DNS on Reload	Tells the Traffic Server to re-resolve hostnames whenever clients reload pages.
	DNS Configuration
Resolve attempt timeout	Specifies how long the Traffic Server must wait for the DNS server to respond with an IP address, even if the client request has been cancelled. Note: If the client abandons the request before this timeout expires, the Traffic Server can still obtain the host's IP address in order to cache it. The next time a client makes the same request, the address will be in the cache. To provide DNS services, the Traffic Server uses a list of DNS servers that it obtains from the DNS table in your `resolv.conf` file. The Traffic Server always tries to connect to the first DNS server on the list; if it cannot make or maintain the connection, it tries the next server on the list.
Number of retries	Specifies how many times the Traffic Server should allow a look-up to fail before it abandons the look-up and sends an `invalid host name` message to the user.

The Logging page

The following table describes the configuration options on the **Logging** page.

Options	Description
	Event Logging
Logging Enabled	Enables/disables the logging feature so that transactions are recorded into event logs and error logs.
Log Transactions Only	Configures Traffic Server to log transactions into your selected event log files only. Errors are not recorded in the error log files.
Log Errors Only	Configures Traffic Server to log errors in the error log files only. Transactions are not logged in event log files.
Logging Disabled	Disables the logging feature.
	Log Management
Log directory	Specifies the path of the directory in which to store event logs. The path of this directory must be the same on every node in the Traffic Server cluster.
Log space limit (MB)	Specifies the maximum amount of space, in MB, allocated to the logging directory for the log files. Note: Transaction logs can consume a lot of space. You should set this limit high enough to accommodate at least a single day's worth of uncompressed transaction logs. Also, make sure that this limit is smaller than the actual space available on the partition that contains the logging directory.
Log space headroom (MB)	Specifies the tolerance for the log space limit. If autodeletion of old log files is enabled, autodeletion is triggered when the amount of free space available in the logging directory is less than the headroom.
	Log Collation
Inactive	Disables log collation. If this option is selected, Traffic Server is neither a collation server nor a collation client.
Be a collation host	Enables Traffic Server to be a log collation host. Since this host will act as the server, there is no need to enter a value for the **to collation host** field.
Send standard formats	Enables Traffic Server to be a log collation client. Selecting this option instructs Traffic Server to send the active standard formats, such as Squid, Netscape Common, etc., to the log collation server. You must enter the name of this collation server for your cluster in the **to collation host** field. Note: When logs are collated, the source of the log entry—its node of origin—is lost unless you turn on the *Log collation host tagged* option (described below). Log collation consumes cluster bandwidth in sending all log entries to a single node. It can therefore impact the performance of the cluster.
Send custom non-xml formats	Enables Traffic Server to be a log collation client. Selecting this option instructs Traffic Server to send traditional custom formats to the specified host. You must enter the name of this collation server for your cluster in the **to collation host** field. Traditional custom formats are specified using the `logs.config` file. For more information, see *Using custom formats, on page 161*.
Send standard and custom non-xml formats	Enables Traffic Server to be a log collation client. Selecting this option instructs Traffic Server to send both standard and traditional custom formats to the specified host. You must enter the name of this collation server for your cluster in the **to collation host** field. This option is the same as having the two previous options enabled at the same time.

Options	Description
Log collation port	Specifies the TCP port identifier that all nodes in the cluster use to exchange event log entries. You must specify a port number in all cases except when log collation is inactive. The default port number is 8085. Note: You should not change the port number unless there is a conflict with another service already using the port.
Log collation secret	Specifies the password for the log collation server and the other nodes in the cluster used to validate logging data and prevent the exchange of arbitrary information.
Log collation host tagged: yes/no	Select Yes to add the hostname of the Traffic Server node that generated the log entry to end of the entry in the collated log file.
Log space limit for orphan log files (MB)	If log collation is enabled, this is the maximum amount of space, in MB, allocated to the logging directory for storing orphan log files on the Traffic Server nodes. Orphan log entries are created when it is not possible to contact the log collation server.
Standard Event Log Formats	
Enabled (On/Off)	Turns event logging in the specified format on or off.
Log file type	Specifies the type of log file: ASCII or binary.

Log file name	Specifies the name of the log file that will record transactions using the selected format style. The default file names are:
	`squid.log`
	`common.log`
	`extended.log`
	`extended2.log`

Log file header	Specifies the text header you want your standard log files to contain.
Log Splitting	
NNTP Log splitting	When the NNTP log splitting is off, Traffic Server records NNTP entries in the same log file with HTTP and FTP entries. When the NNTP log splitting is on, Traffic Server records the NNTP entries in a separate log file.
ICP Log Splitting	When ICP log splitting is off, Traffic Server records ICP entries in the same log file with HTTP and FTP entries. When ICP log splitting is on, Traffic Server records ICP entries in a separate log file.
HTTP Host Log Splitting	If Traffic Server is running in reverse proxy mode, it can log the transactions for each mapped origin server in a separate log file. This feature is called host log splitting. When HTTP host splitting is on, Traffic Server creates a separate log file for each of the hosts listed in the `log_hosts.config` file. When HTTP host splitting is off, Traffic Server records transactions for all hosts in the same log file.
Custom Logs	
Enabled	Enables or disables custom logging.
Custom log definition format	Selecting the Traditional option instructs Traffic Server to look to the `logs.config` file in the config directory for custom log formats. For more information, see *Using custom formats, on page 161*. Selecting XML instructs Traffic Server to look to the `logs_xml.config` file for highly configurable custom log formats. For more information, see *Using custom formats, on page 161*.

Options	Description
	Log File Rolling
Rolling enabled (On/Off)	Enables or disables log file rolling. To keep log files down to manageable sizes, you can roll them at regular intervals. When the Traffic Server rolls a log file, it stops making entries in the file and adds the file name extension `.old`. The log file autodeletion feature deletes rolled log files when space allocated to logging is nearly full (when free space in the logging directory is less than the headroom).
Roll offset hour	Specifies the hour when log rolling takes place. You can set a time of the day in the range 0 to 23. For example, if the offset hour is 0 (midnight) and the roll interval is 6, the log files are rolled at 00:00, 06:00, noon, and 18:00.
Roll interval	Specifies the amount of time the Traffic Server enters data in log files before rolling them to `.old` files. Choose a value from 15 minutes to 24 hours. The minimum value is 5 minutes.
Auto-delete rolled log files when space is low	Enables autodeletion of rolled log files when available space in the log directory is low. Autodeletion is triggered when the amount of free space available in the log directory is less than the headroom specified in the **Log Management** section of the **Logging** page.

The Snapshots page

The following table describes the configuration options on the **Snapshots** page.

Option	Description
Name New Snapshot Take Snapshot	Takes a snapshot and saves a copy of all Traffic Server configuration files. You must first specify a name for the snapshot, then click the **Take Snapshot** button. The name cannot contain a forward slash (/). Note: It is a good idea to take a snapshot before performing system maintenance or attempting to tune system performance. Taking a snapshot only takes a few seconds and it can save you hours of correcting configuration mistakes.
Restore Snapshot	Restores a previously created snapshot so that you can return to a set of configuration values you saved. Select from the list of all the snapshots you have taken, then click **Restore**.
Delete Snapshot	Deletes an existing snapshot. Select the snapshot from the drop-down list, then click the **Delete Snapshot** button.

The Plugins page

The **Plugins** page lists the plugins currently running on your Traffic Server that you can configure using the Traffic Manager UI. A plugin is a program that extends the functionality of Traffic Server. For example, plugins can perform origin server blacklisting, web content filtering, user authentication, and data transformation.

To configure a plugin, click the plugin in the list. A configuration page for the plugin opens.

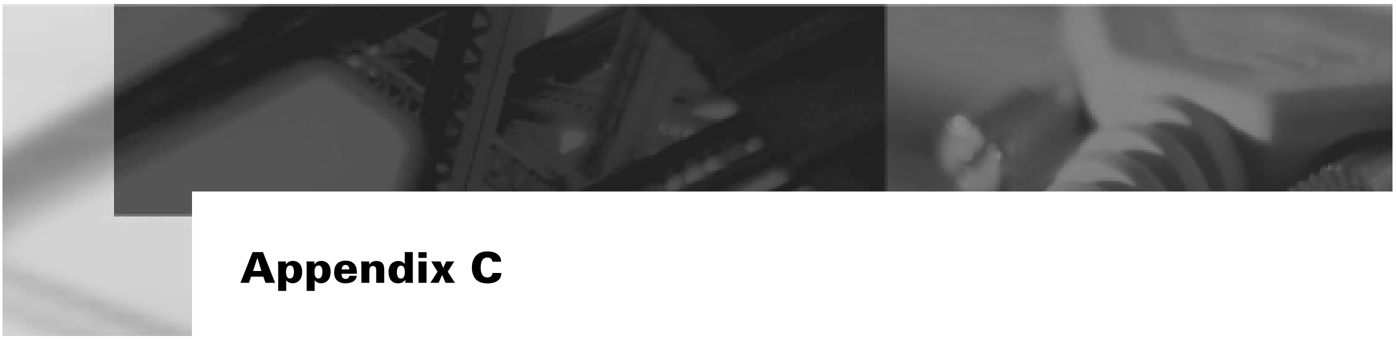
If no plugins configurable from the Traffic Manager UI are running on your Traffic Server, the message `No configurable plugins loaded by Traffic Server` displays on the **Plugins** page.

If you are interested in developing a plugin, refer to the Traffic Server Software Development Kit (SDK) at http://www.inktomi.com/products/network/partners/.

The Content Management page

The following table describes the configuration options on the **Content Management** page.

Option	Description
Scheduled Update	Enables or disables the scheduled update option. When this option is enabled, Traffic Server can automatically update certain objects in the local cache at a specified time.
Update error retry count	Specifies the number of times to retry the scheduled update of a URL in the event of failure.
Update error retry interval (seconds)	Specifies the delay in seconds between each scheduled update retry for a URL in the event of failure.
Maximum concurrent updates	Specifies the maximum simultaneous update requests allowed at any point in time. This option enables you to prevent the scheduled update process from overburdening the host.
URL Update List	Click this link to display the **URL Update List** page enabling you to view, modify, or delete URL entries and their associated Request Headers, update offset hour, interval, and recursion depth. These entries are recorded in the `update.config`. file. See *update.config, on page 297*. The **URL Update List** page contains an **Add Entry** button that opens the **Add Entry** page so that you can add new URLs for which you want Traffic Server to perform a scheduled update to the local cache content. Use the **Add Entry** page to modify settings in the `update.config` file. This is preferable to modifying the file directly using a text editor because the page takes care of automatically notifying Traffic Server when changes are made. If you do modify `update.config` directly, you must manually stop and restart the Traffic Server process. The **Add Entry** page contains the following options: **URL** - HTTP and FTP-based URLs. The system validates the syntax of the URL, but does not confirm its existence. **Request Headers** - (Optional) A semicolon separated list of headers passed in each `GET` request. You can define any request header that conforms to the HTTP specification. The default is no request header. **Update Offset Hour** - The base hour used to derive the update periods. The range is 00-23 hours. **Update Interval** - The interval, in seconds, at which updates should occur, starting at Offset hour. **Recursion Depth** - The depth to which referenced URLs are recursively updated, starting at the given URL. For example, a recursion depth of 1 will update the given URL, as well as all URLs immediately referenced by links from the original URL.
Force immediate update	Overrides the scheduling expiration time for all scheduled update entries and initiates updates until this option is turned off.



Appendix C

Traffic Line Commands

This appendix contains the following sections:

- *Traffic Line batch mode commands, on page 214*
- *Traffic Line interactive mode commands, on page 215*
- *Traffic Line variables, on page 216*

Traffic Line batch mode commands

Use batch mode to execute individual Traffic Server commands. You can also script multiple batch mode commands in a shell.

You execute batch mode commands from Traffic Server's `bin` directory. (In Windows, use a command prompt window.)

The following table describes all the commands available in batch mode.

Command	Description
`traffic_line -i`	Starts command line interactive mode, which lets you view Traffic Server performance and network traffic statistics, and configure the Traffic Server system.
`traffic_line -p socket_`*`path`*	Specifies the location (directory and path) of the file used for Traffic Line and Traffic Manager communication. The default path is *`install_dir`*`/config/cli`.
`traffic_line -r` *`variable`*	Displays specific performance statistics or a current configuration setting. For a list of the variables you can specify, refer to *Traffic Line variables, on page 216*.
`traffic_line -s` *`variable`* `-v` *`value`*	Sets configuration variables. `variable` is the configuration variable you want to change and `value` is the value you want to set. See *page 220* for a list of the configuration variables you can specify.
`traffic_line -h`	Displays the list of Traffic Line commands.
`traffic_line -x`	Initiates a Traffic Server configuration file reread. Executing this command is similar to clicking the **Make These Changes** button in the Traffic Manager UI. Use this command after every configuration file modification.
`traffic_line -M`	Restarts the `traffic_manager` process on all the nodes in a cluster.
`traffic_line -L`	Restarts the `traffic_manager` process on the local node.
`traffic_line -S`	Shuts down the Traffic Server on the local node.
`traffic_line -U`	Starts the Traffic Server on the local node.
`traffic-line -B`	Bounces the Traffic Server cluster wide. Bouncing the Traffic Server shuts down and immediately restarts Traffic Server node by node.
`traffic_line -b`	Bounces the Traffic Server on the local node. Bouncing the Traffic Server shuts down and immediately restarts the Traffic Server node.
`traffic_line -T` *`value`*	Sets the amount of time that Traffic Line waits to receive a response to a command from the Traffic Server. *`value`* specifies the time in seconds.

Traffic Line interactive mode commands

Use interactive mode to retrieve statistics and to configure the Traffic Server system.

To access interactive mode, go to Traffic Server's `bin` directory and enter `traffic_line -i` at the prompt. (In Windows, use a command prompt window.)

The following table describes the commands available in interactive mode.

Note If there is a number associated with a command, enter the number only at the prompt. For example, to access Monitor mode, enter `1` at the prompt.

Command (name)	Description
1 (monitor)	Displays the Monitor mode commands so that you can view Traffic Server performance and network traffic statistics. Refer to *Chapter 9, Monitoring Traffic*.
2 (configure)	Displays the Configure mode commands so that you can configure the Traffic Server system. Refer to *Chapter 10, Configuring Traffic Server*.
3 (reread)	Re-reads the configuration files. Note: Some configuration changes require that you initiate a configuration re-read for the changes to take effect.
4 (shutdown)	Shuts down the Traffic Server.
5 (startup)	Starts the Traffic Server.
6 (bounce_local)	Bounces the Traffic Server on the local node. Bouncing the Traffic Server shuts down and immediately restarts the local node.
7 (bounce_cluster)	Bounces the Traffic Server cluster wide. Bouncing the Traffic Server cluster shuts down and immediately restarts the Traffic Server node by node.
8 (restart _local)	Restarts the `traffic_manager` process on the local node.
9 (restart _cluster)	Restarts the `traffic_manager` process on all the nodes in the cluster.
help	Displays a list of commands for the active level. You can use the help command at any Traffic Line interactive level. Enter either `help` or `?` at the prompt to open the command list.
exit	Exits interactive mode.
.	Moves back to the previous command level.

Traffic Line variables

You can view statistics and change configuration options in Traffic Line by using specific variables. The variables used for gathering statistics are described below. The variables used for viewing and changing configuration options are described in *Configuration Options, on page 220*. For procedures on how to specify the variables, refer to *Viewing Statistics from Traffic Line, on page 117* and *Configuring Traffic Server using Traffic Line, on page 128*.

Statistics

The following table lists the variables you can specify in Traffic Line interactive mode or batch mode to view individual statistics. For a detailed description of the statistics, refer to *Appendix A, Traffic Manager Statistics*.

To view a statistic in batch mode, enter the command `traffic_line -r` *`variable`* at the prompt. In interactive mode, enter the command `get` *`variable`* at the prompt.

Statistic	**Variable**
	Dashboard
Node name	`proxy.node.hostname`
Objects Served	`proxy.node.user_agents_total_documents_served`
Transactions per second	`proxy.node.user_agent__exacts_per_second`
	Node
Cache	
Document Hit Rate	`proxy.node.cache_hit_ratio_avg_10s` `proxy.cluster.cache_hit_ratio_avg_10s`
Bandwidth Savings	`proxy.node.bandwidth_hit_ratio_avg_10s` `proxy.cluster.bandwidth_hit_ratio_avg_10s`
Cache percent free	`proxy.node.cache.percent_free` `proxy.cluster.cache.percent_free`
In Progress	
Open Origin Server Connections	`proxy.node.current_server_connections` `proxy.cluster.current_server_connections`
Open Client Connections	`proxy.node.current_client_connections` `proxy.cluster.current_client_connections`
Cache Xfers In Progress	`proxy.node.current_cache_connections` `proxy.cluster.current_cache_connections`
Network	
Client Throughput (MBits/Sec)	`proxy.node.client_throughput_out` `proxy.cluster.client_throughput_out`
Transactions Per Second	`proxy.node.http.user_agent_xacts_per_second` `proxy.cluster.http.user_agent_xacts_per_second`
Name Resolution	
DNS Lookups Per Second	`proxy.node.dns.lookups_per_second` `proxy.cluster.dns.lookups_per_second`
HostDB Hit Rate	`proxy.node.hostdb.hit_ratio_avg_10s` `proxy.cluster.hostdb.hit_ratio_avg_10s`

Protocols	
HTTP (client)	
Total Document Bytes	`proxy.process.http.user_agent_response_document_total_size`
Total Header Bytes	`proxy.process.http.user_agent_response_header_total_size`
Total Connections	`proxy.process.http.current_client_connections`
Transactions In Progress	`proxy.process.http.current_client_transactions`
HTTP (origin server)	
Total Document Bytes	`proxy.process.http.origin_server_response_document_total_size`
Total Header Bytes	`proxy.process.http.origin_server_response_header_total_size`
Total Connections	`proxy.process.http.current_server_connections`
Transactions In Progress	`proxy.process.http.current_server_transactions`
FTP	
Currently Open Connections	`proxy.process.ftp.connections_currently_open`
Successful PASV Connections	`proxy.process.ftp.connections_successful_pasv`
Unsuccessful PASV Connections	`proxy.process.ftp.connections_failed_pasv`
Successful PORT Connections	`proxy.process.ftp.connections_successful_port`
Unsuccessful PORT Connections	`proxy.process.ftp.connections_failed_port`
ICP (Queries originating from this node)	
Query requests	`proxy.process.icp.icp_query_requests`
Query messages sent	`proxy.process.icp.total_udp_send_queries`
Peer hit messages received	`proxy.process.icp.icp_query_hits`
Peer miss messages received	`proxy.process.icp.icp_query_misses`
Total responses received	`proxy.process.icp.icp_remote_responses`
Average ICP message response time (ms)	`proxy.process.icp.total_icp_response_time`
Average ICP request time (ms)	`proxy.process.icp.total_icp_request_time`
ICP (Queries originating from ICP Peers)	
Query messages received	`proxy.process.icp.icp_remote_query_requests`
Remote query hits	`proxy.process.icp.cache_lookup_success`
Remote query misses	`proxy.process.icp.cache_lookup_fail`
Successful response messages sent to peers	`proxy.process.icp.query_response_write`
NNTP (client)	
Open Connections	`proxy.process.nntp.client_connections_currently_open`
Bytes Read	`proxy.process.nntp.client_bytes_read`
Bytes Written	`proxy.process.nntp.client_bytes_written`
NNTP (server)	
Open Connections	`proxy.process.nntp.server_connections_currently_open`
Bytes Read	`proxy.process.nntp.server_bytes_read`
Bytes Written	`proxy.process.nntp.server_bytes_written`
NNTP (operations)	
Article Hits	`proxy.process.nntp.article_hits`
Article Misses	`proxy.process.nntp.article_misses`

Overview Hits	`proxy.process.nntp.overview_hits`
Overview Refreshes	`proxy.process.nntp.overview_refreshes`
Group Hits	`proxy.process.nntp.group_hits`
Group Refreshes	`proxy.process.nntp.group_refreshes`
Posts	`proxy.process.nntp.posts`
Post Bytes	`proxy.process.nntp.post_bytes`
Pull Bytes	`proxy.process.nntp.pull_bytes`
Feed Bytes	`proxy.process.nntp.feed_bytes`
WCCP Router statistics	
Router's IP address	`proxy.node.wccp.router_ip`
Router status	`proxy.node.wccp.router_status`
WCCP Node statistics	
Node IP address	`proxy.node.wccp.my.ip`
Percentage of traffic received	`proxy.node.wccp.my_share`
Number of heartbeats	`proxy.node.wccp.hbeats_received`
WCCP Protocol statistics	
Enabled	`proxy.node.wccp.enabled`
Leader's IP address	`proxy.node.wccp.leader_ip`
Number of active nodes	`proxy.node.wccp.number_of_caches_up`
Cache	
Bytes Used	`proxy.process.cache.bytes_used`
Cache Size	`proxy.process.cache.bytes_total`
Lookups in Progress	`proxy.process.cache.lookup.active`
Lookups Completed	`proxy.process.cache.lookup.success`
Lookup Misses	`proxy.process.cache.lookup.failure`
Reads in Progress	`proxy.process.cache.read.active`
Reads Completed	`proxy.process.cache.read.success`
Read Misses	`proxy.process.cache.read.miss`
Writes in Progress	`proxy.process.cache.write.active`
Writes Completed	`proxy.process.cache.write.success`
Write Failures	`proxy.process.cache.write.cancel`
Updates in Progress	`proxy.process.cache.update.active`
Updates Completed	`proxy.process.cache.update.success`
Update Failures	`proxy.process.cache.update.failure`
Removes in Progress	`proxy.process.cache.remove.active`
Remove Successes	`proxy.process.cache.remove.success`
Remove Failures	`proxy.process.cache.remove.failure`
Other	
HostDB	
Total Lookups	`proxy.process.hostdb.total_lookups`
Total Hits	`proxy.process.hostdb.total_hits`
Time TTL (min)	`proxy.process.hostdb.ttl`

DNS	
DNS Total Look Ups	`proxy.process.dns.total_dns_lookups`
Average Lookup Up Time (ms)	`proxy.process.dns.lookup_avg_time`
DNS Successes	`proxy.process.dns.lookup_successes`
	Cluster
Bytes Read	`proxy.process.cluster.read_bytes`
Bytes Written	`proxy.process.cluster.write_bytes`
Connections Open	`proxy.process.cluster.connections_open`
Total Operations	`proxy.process.cluster.connections_opened`
Network Backups	`proxy.process.cluster.net_backup`
Clustering Nodes	`proxy.process.cluster.nodes`
	SOCKS
Connections Unsuccessful	`proxy.process.socks.connections_unsuccessful`
Successful Connections	`proxy.process.socks.connections_successful`
Connections in progress	`proxy.process.socks.connections_currently_open`
	Logging
Currently Open Log Files	`proxy.process.log2.log_files_open`
Space Used For Log Files	`proxy.process.log2.log_files_space_used`
Number of Access Events Logged	`proxy.process.log2.event_log_access`
Number of Access Events Skipped	`proxy.process.log2.event_log_access_skip`
Number of Error Events Logged	`proxy.process.log2.event_log_error`

Configuration Options

The following table lists the variables you can specify in Traffic Line interactive mode or batch mode to configure Traffic Server. For a description of the configuration options, refer to *Appendix B, Traffic Manager Configuration Options*.

Configuration Option	Variable
	Server
Traffic Server Name	`proxy.config.proxy_name`
Traffic Server Port	`proxy.config.http.server_port`
Local Domain Expansion	`proxy.config.dns.search_default_domains` 1 = Enable 0 = Disable
.com Domain Expansion	`proxy.config.http.enable_url_expandomatic` 1 = Enable 0 = Disable
Web Management	
Traffic Manager Port	`proxy.config.admin.web_interface_port`
Refresh rate in Monitor mode	`proxy.config.admin.ui_refresh_rate`
Virtual IP Addressing	
Virtual IP	`proxy.config.vmap.enabled` 1 = Enable 0 = Disable
Auto Configuration	
Auto-configuration Port	`proxy.config.admin.autoconf_port`
Throttling of Network Connections	
Maximum Number of Connections	`proxy.config.net.connections_throttle`
SNMP	
SNMP Master Agent	`proxy.config.snmp.master_agent_enabled` 1 = Enable 0 = Disable
Customizable Response Pages	
Suppress generated response pages	`proxy.config.body_factory.response_suppression_mode` 0 = never 1 = always 2 = only when transparent
Enable Custom Response Pages	`proxy.config.body_factory.enable_customizations` 0 = disable 1 = enable default custom pages 2 = enable language-targeted custom pages
Log Customization Activity to Error Log	`proxy.config.body_factory.enable_logging` 1 = Enable 0 = Disable
Custom Response Page Template Directory	`proxy.config.body_factory.template_sets_dir`

Configuration Option	Variable
	Protocols
HTTP	
Keep-Alive Timeout: Inbound (secs)	`proxy.config.http.keep_alive_no_activity_timeout_in`
Keep-Alive Timeout: Outbound (secs)	`proxy.config.http.keep_alive_no_activity_timeout_out`
Inactivity Timeout: Inbound (secs)	`proxy.config.http.transaction_no_activity_timeout_in`
Inactivity Timeout: Outbound (secs)	`proxy.config.http.transaction_no_activity_timeout_out`
Activity Timeout: Inbound (secs)	`proxy.config.http.transaction_active_timeout_in`
Activity Timeout: Outbound (secs)	`proxy.config.http.transaction_active_timeout_out`

Remove HTTP headers:From
Remove HTTP headers: Referer
Remove HTTP headers: User Agent
Remove HTTP headers: Cookie
	`proxy.config.http.anonymize_remove_from`
	`proxy.config.http.anonymize_remove_referer`
	`proxy.config.http.anonymize_remove_user_agent`
	`proxy.config.http.anonymize_remove_cookie`
	`1` = Yes
	`0` = No

Comma-separated list of headers to remove	`proxy.config.http.anonymize_other_header_list`
Insert Client-IP headers	`proxy.config.http.anonymize_insert_client_ip` `1` = Yes `0` = No
Remove Client-IP headers	`proxy.config.http.anonymize_remove_client_ip` `1` = Yes `0` = No
HTTPS	
Restrict SSL connections to port	`proxy.config.http.ssl_ports`
FTP	
FTP connection mode:	`proxy.config.ftp.data_connection_mode` `1` = PASV/PORT (use PORT if PASV fails) `2` = PASV only (initiate data connection) `3` = PORT only (receive data connection)
FTP inactivity timeout	`proxy.config.ftp.control_connection_timeout`
Anonymous FTP password	`proxy.config.http.ftp.anonymous_passwd`
NNTP	
NNTP Server enabled	`proxy.config.nntp.enabled` `1` = Enable `0` = Disable
NNTP Server Port	`proxy.config.nntp.server_port`
Posting enabled	`proxy.config.nntp.posting_enabled` `1` = Enable `0` = Disable
Access Control enabled	`proxy.config.nntp.access_control_enabled` `1` = Enable `0` = Disable

Configuration Option	Variable
NNTP V2 Authentication enabled	`proxy.config.nntp.v2_authentication` 1 = Enable 0 = Disable
Run Local Authentication Server	`proxy.config.nntp.run_local_authentication_server`
Clustering enabled	`proxy.config.nntp.cluster_enabled`
Feed enabled	`proxy.config.nntp.feed_enabled` 1 = Enable 0 = Disable
Logging enabled	`proxy.config.nntp.logging_enabled` 1 = Enable 0 = Disable
Background Posting enabled	`proxy.config.nntp.background_posting_enabled` 1 = Enable 0 = Disable
Obey Cancel Control Messages	`proxy.config.nntp.obey_control_cancel`
Obey NewGroups Control Messages	`proxy.config.nntp.obey_control_newgroup`
Obey RmGroups Control Messages	`proxy.config.nntp.obey_control_rmgroup`
Inactivity Timeout(secs)	`proxy.config.nntp.inactivity_timeout`
Check for New Groups Every(secs)	`proxy.config.nntp.check_newgroups_every`
Check for Cancelled Articles Every(secs):	`proxy.config.nntp.check_cancels_every`
Check Parent NNTP Server Every(secs):	`proxy.config.nntp.group_check_parent_every`
Check Cluster Every(secs)	`proxy.config.nntp.group_check_cluster_every`
Check Pull Groups Every(secs):	`proxy.config.nntp.check_pull_every`
Authorization Server Host	`proxy.config.nntp.authorization_hostname`
Authorization Server Port	`proxy.config.nntp.authorization_port`
Authorization Server Timeout(mill-secs):	`proxy.config.nntp.authorization_server_timeout`
Client Speed Throttle(bytes/secs):	`proxy.config.nntp.client_speed_throttle`
Cache	
Activation	
Enable HTTP caching	`proxy.config.http.cache.http` 1 = Enable 0 = Disable
Enable FTP caching	`proxy.config.http.cache.ftp` 1 = Enable 0 = Disable
Enable NNTP caching	`proxy.config.nntp.cache.enabled` 1 = Enable 0 = Disable
Ignore user requests to bypass cache	`proxy.config.http.cache.ignore_server_no_cache` 1 = Yes 0 = No

Configuration Option	Variable
Storage	
Maximum number of alternates allowed for a URL	`proxy.config.cache.limits.http.max_alts`
Freshness	
Verify freshness by checking when object has expired, when object has expired or has no expiration date, always, or never	`proxy.config.http.cache.when_to_revalidate` `0` = when the object has expired `1` = when the object has expired, or has no expiration date `2` = always `3` = never
Minimum information needed to cache document	`proxy.config.http.cache.required_headers` `0` = nothing `1` = a last-modified time `2` = an explicit lifetime
minimum life time (secs)	`proxy.config.http.cache.heuristic_min_lifetime`
maximum life time (secs)	`proxy.config.http.cache.heuristic_max_lifetime`
FTP cached objects expire after (secs)	`proxy.config.http.ftp.cache.document_lifetime`
Variable Content	
Do not cache objects served in response to URLs that contain a question mark, a semi-colon, cgi, or end in asp.	`proxy.config.http.cache.cache_urls_that_look_dynamic`
Do not cache objects served in response to URLs that contain cookies	`proxy.config.http.cache.cache_responses_to_cookies`
Enable Alternates	`proxy.config.http.cache.enable_default_vary_headers` `1` = Enable `0` = Disable

If alternates are enabled, then vary on these headers:
If the request is for text
If the request is for images
If the request is for anything else
	
	`proxy.config.http.cache.vary_default_text`
	`proxy.config.http.cache.vary_default_images`
	`proxy.config.http.cache.vary_default_other`

Security	
Access	
Authentication (basic):	`proxy.config.admin.basic_auth` `1` = Enable `0` = Disable
Administrator's ID	`proxy.config.admin.admin_user`
Administrator's Password	`proxy.config.admin.admin_password`
Firewall Configuration	
SOCKS	`proxy.config.socks.socks_needed` `1` = Enable `0` = Disable
SOCKS server IP address	`proxy.config.socks.socks_server_ip_str`
SOCKS server port	`proxy.config.socks.socks_server_port`

Configuration Option	Variable
SOCKS timeout (seconds)	`proxy.config.socks.socks_timeout`
	Logging
Event Logging	
Event Logging	`proxy.config.log2.logging_enabled` `0` = no logging at all `1` = log errors only `2` = full logging
Log Management	
Log directory	`proxy.config.log2.logfile_dir`
Log space limit (MB)	`proxy.config.log2.max_space_mb_for_logs`
Log space Headroom(MB)	`proxy.config.log2.max_space_mb_headroom`
Log Collation	
Log collation host	`proxy.config.log2.collation_host`
Log collation port	`proxy.config.log2.collation_port INT`
Log collation secret	`proxy.config.log2.collation_secret`
Log space limit for orphan log files (MB)	`proxy.config.log2.max_space_mb_for_orphan_logs`
Squid Format	
Squid Enabled	`proxy.config.log2.squid_log_enabled` `1` = Enable `0` = Disable
Squid Log file type	`proxy.config.log2.squid_log_is_ascii` `1` = ASCII `0` = binary
Squid Log file name	`proxy.config.log2.squid_log_name`
Log file header	`proxy.config.log2.squid_log_header`
Netscape Common Format	
Netscape Common Enabled	`proxy.config.log2.common_log_enabled` `1` = Enable `0` = Disable
Netscape Common Log file type	`proxy.config.log2.common_log_is_ascii` `1` = ASCII `0` = binary
Netscape Common Log file name	`proxy.config.log2.common_log_name`
Netscape Common Log file header	`proxy.config.log2.common_log_header`
Netscape Extended Format	
Netscape Extended Enabled	`proxy.config.log2.extended_log_enabled` `1` = Enable `0` = Disable
Netscape Extended Log file type	`proxy.config.log2.extended_log_is_ascii` `1` = ASCII `0` = binary

Configuration Option	Variable
Netscape Extended Log file name	`proxy.config.log2.extended_log_name`
Netscape Extended Log file header	`proxy.config.log2.extended_log_header`
Netscape Extended2 Format	
Netscape Extended2 Enabled	`proxy.config.log2.extended2_log_enabled` `1` = Enable `0` = Disable
Netscape Extended2 Log file type	`proxy.config.log2.extended2_log_is_ascii` `1` = ASCII `0` = binary
Netscape Extended2 Log file name	`proxy.config.log2.extended2_log_name`
Netscape Extended2 Log file header	`proxy.config.log2.extended2_log_header`
Custom logs enabled	`proxy.config.log2.custom_logs_enabled` `1` = Enable `0` = Disable
Log File Rolling	
Rolling Enabled	`proxy.config.log2.rolling_enabled` `1` = Enable `0` = Disable
Roll offset hour(24hr):	`proxy.config.log2.rolling_offset_hr`
Roll interval(sec)	`proxy.config.log2.rolling_interval_sec`
Auto-delete rolled log files when space is low	`proxy.config.log2.auto_delete_rolled_files`
Log Splitting	
NNTP Log Splitting	`proxy.config.log2.separate_nntp_logs` `1` = Enable `0` = Disable
Host Log Splitting	`proxy.config.log2.separate_host_logs` `1` = Enable `0` = Disable
	Routing
Parent Proxy	
Parent Caching	`proxy.config.http.parent_proxy_routing_enable` `1` = Enable `0` = Disable
Parent Cache:	`proxy.config.http.parent_proxies`
ICP	
ICP mode:	`proxy.config.icp.enabled` `0` = Disable `1` = Only Receive Queries `2` = Send/Receive Queries
ICP Port	`proxy.config.icp.icp_port`

Configuration Option	Variable
ICP multicast enabled	`proxy.config.icp.multicast_enabled` `1` = Enable `0` = Disable
ICP Query Timeout	`proxy.config.icp.query_timeout`
Reverse Proxy	
Server Acceleration	`proxy.config.reverse_proxy.enabled` `1` = Enable `0` = Disable
Require Document Route Rewriting	`proxy.config.url_remap.remap_required` `1` = Yes `0` = No
URL to redirect requests without Host header	`proxy.config.url_remap.pristine_host_hdr`
HostDB	
Host Database Management	
Lookup timeout(secs)	`proxy.config.hostdb.lookup_timeout`
Foreground timeout(secs)	`proxy.config.hostdb.timeout`
Background timeout(secs)	`proxy.config.hostdb.verify_after`
Invalid host timeout (minutes)	`proxy.config.hostdb.fail.timeout`
Re-DNS on Reload	`proxy.config.hostdb.re_dns_on_reload` `1` = Enable `0` = Disable
Dns Configuration	
Resolve attempt timeout(secs)	`proxy.config.dns.lookup_timeout`
Number of retries	`proxy.config.dns.retries`

Appendix D

Configuration Files

This appendix describes the Traffic Server configuration files that you can edit. For a list of all the configuration files used by Traffic Server, including files that you must not edit, refer to the *Traffic Server Installation Guide*.

arm_security.config

The `arm_security.config` file contains the ARM access control list. The file consists of a series of open, allow, and deny lines that specify the hosts that are allowed to communicate with the Traffic Server ARM using TCP and UDP through defined ports. Traffic Server uses this configuration file when the ARM security option is enabled. For information about enabling the ARM security option, refer to *Controlling host access to the Traffic Server machine (ARM security), on page 134*.

By default, the `arm_security.config` file adopts a strategy whereby all ports except for 8080, used by Traffic Server, are closed unless explicitly opened. This means that when using this configuration file, you must open the ports that are being used by Traffic Server, among others. Otherwise remote origin servers will be unable to communicate with the proxy.

Lines within the `arm_security.config` generally assume the following functional order:

✔ Define the ports that are to be open by default, for either TCP and UDP

✔ Define the hosts that are to be denied access to specific destination ports, for either TCP and UDP

✔ Define the hosts that are to be allowed access to specific destination ports, for either TCP or UDP

Caution Before you enable the ARM security option, ensure that you have either console access to the Traffic Server machine or that you have added the appropriate rules to the configuration file to allow telnet or ssh access for yourself.

Format

Each line in the `arm_security.config` file uses one of the following formats:

```
open tcp | udp ports o_ports
```

```
deny tcp | udp dport d_ports src src_IPaddresses
```

```
allow tcp | udp src src_IPaddresses dst dst_IPaddresses dport d_ports
sport s_ports
```

The following table describes each field:

Field	Allowed inputs
o_ports	The port, or series of ports separated by spaces, to open by default.
d_ports	The destination port, or series of destination ports separated by spaces, through which TCP traffic should either be allowed or denied.
s_ports	The source port, or series of source ports separated by spaces, from which TCP traffic should be allowed.
src_IPaddresses	The IP address, or range of IP addresses, specifying the source of the communication.
dst_IPaddresses	The IP address, or range of IP addresses, specifying the destination of the communication.

Examples

The following example defines ports 80, 119, 23, and 554 as open for TCP communication. All other ports are closed:

```
open tcp ports 80 119 23 554
```

In the following example, the first line specifies that all hosts are denied access to destination port 80 using TCP. The second line specifies that host 209.1.2.2 is denied access to destination port 90 using UDP:

```
deny tcp dport 80 src 0.0.0.0-255.255.255.255
```

```
deny udp dport 90 src 209.1.2.2
```

In the following example, the first line specifies that host 1.1.1.1 using source port 20 is allowed to communicate with host 5.5.5.5 on destination ports 127-130 using TCP. The second line specifies that all hosts are allowed to communicate with host 1.1.2.4 using UDP:

```
allow tcp src 1.1.1.1 dst 5.5.5.5 dport 127-130 sport 20
```

```
allow udp dst 1.1.2.4
```

bypass.config

The `bypass.config` file contains *static* transparency bypass rules (refer to *Static bypass rules, on page 63* for information about using static bypass rules). When the transparency option is enabled, the Traffic Server uses the rules in the `bypass.config` file to determine whether to bypass incoming client requests or attempt to serve them transparently.

You can configure three types of bypass rules:

Rule	Description
Source bypass	Configures the Traffic Server to bypass a particular source IP address or range of IP addresses. For example, use this solution to bypass clients that do not want to use caching.
Destination bypass	Configures the Traffic Server to bypass a particular destination IP address or range of IP addresses. For example, these could be destination servers that use IP authentication based on the client's real IP address. Note: Destination bypass rules prevent the Traffic Server from caching an entire site. You will experience hit rate impacts if the site you bypass is popular.
Source/Destination pair bypass	Configures the Traffic Server to bypass requests that originate from the specified source to the specified destination. For example, you can route around specific client-server pairs that experience broken IP authentication or out-of-band HTTP traffic problems when cached. Source/destination bypass rules can be preferable to destination rules because they block a destination server only for users that experience problems.

The `bypass.config` file also accepts dynamically generated bypass rules. You can configure the Traffic Server to generate destination or source/destination bypass rules in the following instances:

✔ If there is a non-HTTP request on port 80

✔ If an HTTP request returns the following errors:

- ✗ 400 Bad Request error
- ✗ 401 Unauthorized error
- ✗ 403 Forbidden error
- ✗ 405 Method not allowed error
- ✗ 406 Not Acceptable (access) error
- ✗ 408 Request timeout error
- ✗ 500 Internal server error

Format

The bypass rules have the following format:

Rule	Format
source IP bypass	`bypass src` *`IPaddress`* Where *`IPaddress`* can be: - a simple IP address, such as 1.1.1.1 - in CIDR (Classless Inter-Domain Routing) format, such as 1.1.1.0/24 - a range separated by a dash, such as 1.1.1.1-2.2.2.2 - any combination of the above, separated by commas, such as 1.1.1.0/24, 25.25.25.25, 123.1.23.1 - 123.1.23.123
destination IP bypass	`bypass dst` *`IPaddress`* Where *`IPaddress`* has the same format as `src` *`IPaddress`*
source/destination IP bypass	`bypass src` *`IPaddress`* `AND dst` *`IPaddress`* Where *`IPaddress`* must be a single IP address, such as 1.1.1.1

Important After you modify the `bypass.config` file, the Traffic Manager has to reread the configuration files. Make Traffic Server's `bin` directory your working directory, and run the `traffic_line -x` command. If you are running a cluster, you need only run the command for one node; the changes will propagate.

Examples

The following are example source, destination, and source/destination bypass rules:

```
bypass src 1.1.1.0/24, 25.25.25.25, 128.252.11.11 - 128.252.11.255
bypass dst 24.24.24.0/24
bypass src 25.25.25.25 AND dst 24.24.24.0
```

cache.config

The Traffic Server caches objects indexed by URLs. In the `cache.config` file, you can specify how a particular group of URLs should be cached. The `cache.config` file contains caching rules that enable you to specify:

- ✔ Whether to cache objects
- ✔ How long to pin particular objects in the cache
- ✔ How long to consider cached objects as fresh
- ✔ Whether to ignore no-cache directories from the server

Important After you modify the `cache.config` file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's `bin` directory your working directory, then run the `traffic_line -x` command. In Windows, open a Command Prompt window, `cd` to the `bin` directory (located in the Traffic Server installation directory), then run the `traffic_line -x` command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line in the `cache.config` file contains a caching rule. Traffic Server recognizes three space-delimited tags:

```
primary destination=value secondary specifier=value action=value
```

Note You can use more than one secondary specifier in a rule. However, you cannot repeat a secondary specifier.

The following table lists the possible primary destinations and their allowed values:

Primary Destination	Allowed
`dest_domain`	Requested domain name
`dest_host`	Requested hostname
`dest_ip`	Requested IP address
`url_regex`	Regular expression to be found in a URL

The secondary specifiers are optional in the `cache.config` file. The following table lists the possible secondary specifiers and their allowed values:

Secondary Specifier	Allowed Value
`time`	A time range, such as 08:00-14:00
`src_ip`	The IP address of the client
`prefix`	A prefix in the path part of a URL
`suffix`	A file suffix in the URL
`port`	A requested URL port

Secondary Specifier	Allowed Value
`method`	A request URL method; one of the following: get post put trace
`scheme`	A request URL protocol; one of the following: HTTP FTP

The following table lists the possible actions and their allowed values:

Action	Value
`action`	never-cache ignore-no-cache
`pin-in-cache`	Enter the amount of time you want to keep the object(s) in the cache. Following time formats are allowed: h for hours, e.g. 10h m for minutes, e.g. 5m s for seconds, e.g. 20s mixed units, e.g. 1h15m20s
`revalidate`	Enter the amount of time you want to consider the object(s) fresh. Use the same time formats as `pin-in-cache`.

Examples

The following rule configures Traffic Server to never cache FTP documents requested from the IP address 112.12.12.12.

```
dest_ip=112.12.12.12 scheme=ftp action=never-cache
```

The following rule configures Traffic Server to keep documents with URLs that contain the regular expression `politics` and with the path prefix `/viewpoint` in the cache for 12 hours.

```
url_regex=politics prefix=/viewpoint pin-in-cache=12h
```

The following rules configures Traffic Server to revalidate `gif` and `jpeg` objects in the domain `inktomi.com` every 6 hours and all other objects in `inktomi.com` every hour.

Note The rules are applied in the order listed.

```
dest_domain=inktomi.com suffix=gif revalidate=6h
dest_domain=inktomi.com suffix=jpeg revalidate=6h
dest_domain=inktomi.com revalidate=6h
```

filter.config

The `filter.config` file lets you deny or allow particular URL requests and keep or strip header information. Allowing a URL request means that the Traffic Server will cache and serve the requested document. When a request is denied, the client receives an `access denied` message.

Important After you modify the `filter.config` file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's `bin` directory your working directory, then run the `traffic_line -x` command. In Windows, open a Command Prompt window, `cd` to the `bin` directory (located in the Traffic Server installation directory), then run the `traffic_line -x` command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line in the `filter.config` file contains a filtering rule. Traffic Server recognizes three space-delimited tags:

```
primary destination=value secondary specifier=value action=value
```

Note You can use more than one secondary specifier in a rule. However, you cannot repeat a secondary specifier.

The following table lists the possible primary destinations and their allowed values:

Primary Destination	Allowed Value
`dest_domain`	Requested domain name
`dest_host`	Requested hostname
`dest_ip`	Requested IP address
`url_regex`	Regular expression to be found in a URL

The secondary specifiers are optional in the `filter.config` file. The following table lists the possible secondary specifiers and their allowed values:

Secondary Specifiers	Allowed Value
`time`	A time range, such as 08:00-14:00
`src_ip`	The IP address of the client
`prefix`	A prefix in the path part of a URL
`suffix`	A file suffix in the URL
`port`	A requested URL port
`method`	A request URL method; one of the following: get post put trace
`scheme`	A request URL protocol; one of the following: HTTP FTP

The following table lists the possible actions and their allowed values:

Action	Value
`action`	allow deny PUSH - If the PUSH option is enabled (the PUSH option lets you deliver content directly to the cache without user request), you must add a filtering rule with the PUSH action to ensure that only known source IP addresses implement PUSH requests to the cache. See the example below. You enable the PUSH option by setting the configuration variable `proxy.config.http.push_method_enabled` to 1 in the `records.config` file.
`keep_hdr`	Enter the client request header information that you want to keep: date host cookie client_ip
`strip_hdr`	Enter the client request header information that you want to strip. You have the same options as `keep_hdr`.

Examples

The following rule configures Traffic Server to deny FTP document requests to the IP address 112.12.12.12.

```
dest_ip=112.12.12.12 scheme=ftp action=deny
```

The following rule configures Traffic Server to keep the client IP address header for URLs that contain the regular expression `politics` and whose path prefix is `/viewpoint`.

```
url_regex=politics prefix=/viewpoint keep_hdr=client_ip
```

The following rule configures Traffic Server to strip all cookies to the requested host `www.inktomi.com`.

```
dest_host=www.inktomi.com strip_hdr=cookie
```

The following rule configures Traffic Server not to allow `puts` to the requested host `www.inktomi.com`.

```
dest_host=www.inktomi.com method=put action=deny
```

The following rule configures Traffic Server to allow only the host associated with the IP address 11.11.1.1 to deliver content directly into the cache.

Important If you enable the PUSH option, you must add a line similar to the following to prevent unauthorized users from putting content in the cache.

```
dest_domain=. src_ip=11.11.1.1 method=PUSH action=allow
dest_domain=. method=PUSH action=deny
```

ftp_remap.config

The ftp_remap.config file is used for FTP reverse proxy and contains mapping rules that Traffic Server uses to direct any incoming FTP requests to the FTP server if the requested document is a cache miss or is stale.

Important After you modify the ftp_remap.config file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's bin directory your working directory, then run the traffic_line -x command. In Windows, open a Command Prompt window, cd to the bin directory (located in the Traffic Server installation directory), then run the traffic_line -x command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

For information about using and configuring FTP reverse proxy, refer to *Chapter 5, Reverse Proxy and HTTP Redirects*.

Format

Each line in the ftp_remap.config file contains a mapping rule in the format:

```
Traffic_Server_IPaddress:port ftp_server_IPaddress:port
```

where *Traffic_Server_IPaddress* is the IP address assigned to Traffic Server.

ftp_server_IPaddress is the IP address assigned to the FTP server to which you want to redirect requests.

port is the port number.

Examples

In the following example, all FTP requests sent to Traffic Server's IP address of 111.111.11.1 are directed to the FTP server's IP address 11.11.11.1 when a request is a cache miss or is stale.

```
111.111.11.1:7999 11.11.11.1:21
```

hosting.config

The `hosting.config` file lets you assign cache partitions to specific origin servers and/or domains so that you can manage your cache space more efficiently, and restrict disk usage.

For step by step instructions on partitioning the cache according to origin servers and/or domains, refer to *Partitioning the cache according to origin server or domain, on page 105*.

Note Before you can assign cache partitions to specific origin servers and/or domains, you must partition your cache according to size and protocol in the `partition.config` file. For step by step instructions on partitioning your cache, refer to *Partitioning the cache, on page 104*. For a description of the `partition.config` file, refer to *partition.config, on page 257*.

After you modify the `hosting.config` file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's `bin` directory your working directory, then run the `traffic_line -x` command. In Windows, open a Command Prompt window, `cd` to the `bin` directory (located in the Traffic Server installation directory), then run the `traffic_line -x` command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Important The partition configuration must be the same on all nodes in a cluster.

Format

Each line in the `hosting.config` file must have one of the following formats:

```
hostname=hostname  partition=list_of_partition_numbers
```

```
domain=domain_name  partition=list_of_partition_numbers
```

where *`hostname`* is the fully qualified hostname of the origin server whose content you want to store on a particular partition (for example, `www.inktomi.com`).

`domain_name` is the domain whose content you want to store on a particular partition (for example, `inktomi.com`).

`list_of_partition_numbers` is a comma-separated list of the partitions on which you want to store the content that belongs to the origin server or domain listed. The partition numbers must be valid numbers listed in the `partition.config` file (refer to *partition.config, on page 257*).

Generic Partition

When configuring the `hosting.config` file, you must assign a generic partition to use for content that does not belong to any of the origin servers or domains listed. If all partitions for a particular origin server become corrupt, Traffic Server will also use the generic partition to store content for that origin server.

The generic partition must have the following format:

```
hostname=* partition=list_of_partition_numbers
```

where *`list_of_partition_numbers`* is a comma-separated list of generic partitions.

Examples

In the following example, content from the domain `inktomi.com` is stored on partition 1 and content from `www.inktomi.com` is stored on partition 2. Content from all other origin servers is stored in partitions 3 and 4.

```
domain=inktomi.com partition=1
hostname=www.inktomi.com partition=2
hostname=* partition=3,4
```

icp.config

This file defines ICP peers (parent and sibling caches). Refer to *ICP cache hierarchies, on page 95*.

Important After you modify the `icp.config` file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server’s `bin` directory your working directory, then run the `traffic_line -x` command. In Windows, open a Command Prompt window, `cd` to the `bin` directory (located in the Traffic Server installation directory), then run the `traffic_line -x` command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line in the `icp.config` file contains the name and configuration information for a single ICP peer, in the following format:

```
host:host_IP:cache_type:proxy_port:icp_port:MC_on:MC_IP:MC_TTL:
```

Each field is described in the following table:

Field	Description
host	The hostname of the ICP peer. The name `localhost` is reserved for the Traffic Server.
host _IP	The IP address of the ICP peer.
cache _type	Use the following options: 1 to indicate an ICP parent cache 2 to indicate an ICP sibling cache Option 3 is reserved for the local host (the Traffic Server itself).
proxy _port	The port number of the TCP port used by the ICP peer for proxy communication.
icp_port	The port number of the UDP port used by the ICP peer for ICP communication.
MC _on	Multicast on/off. Use the following options: 0 if multicast is not enabled. 1 if multicast is enabled.
MC _IP	The multicast IP address.
MC _TTL	The multicast time to live. Use the following options: 1 if IP multicast datagrams will not be forwarded beyond a single subnetwork 2 to allow delivery of IP multicast datagrams to more than one subnet (if there are one or more multicast routers attached to the first hop subnet)

Examples

The following example configuration is for 3 nodes: the local host, one parent, and one sibling:

```
localhost:0.0.0.0:3:8080:3130:0:0.0.0.0:0:
host1:123.12.1.23:1:8080:3131:0:0.0.0.0:0:
host2:123.12.1.24:1:8080:3131:0:0.0.0.0:0:
```

ip_allow.config

The `ip_allow.config` file controls client access to the Traffic Server proxy cache. You can specify ranges of IP addresses that are allowed to use the Traffic Server as a web proxy. If you want to deny Traffic Server access to specific IP addresses, do not include them in any line in the `ip_allow.config` file.

Important After you modify the `ip_allow.config` file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's `bin` directory your working directory, then run the `traffic_line -x` command. In Windows, open a Command Prompt window, `cd` to the `bin` directory (located in the Traffic Server installation directory), then run the `traffic_line -x` command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line in the `ip_allow.config` file must have the following format:

```
src_ip=IPaddress_or_range_of_IPaddresses action=ip_allow
```

where *`IPaddress_or_range_of_IPaddresses`* is the IP address or range of IP addresses of the clients allowed to access the Traffic Server.

Examples

The following line in the `ip_allow.config` file allows all clients to access the Traffic Server proxy cache:

```
src_ip=0.0.0.0-255.255.255.255 action=ip_allow
```

The following line allows all clients on a specific subnet to access the Traffic Server proxy cache:

```
src_ip=123.12.3.000-123.12.3.123 action=ip_allow
```

ldapsrvr.config

The ldapsrvr.config file enables you to specify sites on the Internet that Traffic Server clients can access without being authenticated by the LDAP server. Using this configuration file, you can also specify the LDAP server that Traffic Server should use for specific objects or sets of objects.

For more information, refer to *Configuring LDAP-based proxy authentication, on page 146*.

Format

Each line in the ldapsrvr.config file consists of a set of tag value pairs. The pairs are in the format *tag=value*. You must include exactly one primary specifier for each line. The following table describes the primary destination specifiers:

Primary Destination	Allowed
dest_domain	A valid domain name. This specifies that the URL selection be based on the destination domain.
dest_host	A valid hostname. This specifies that the URL selection be based on the destination host.
dest_ip	A valid IP address. This specifies that the URL selection be based on the IP address.
url_regex	A valid URL regular expression. This specifies that the URL selection be based on a regular expression.

The following table describes the available server directives:

Server Directive	Description
server	(Required) Must be set to the keyword "auth_bypass" in order to activate the authentication bypass feature. All URLs that match the keyword are exempted from proxy authentication.
dn	Not required when set to "auth_bypass"
uid_filter	(Optional) Defaults to "uid"

Examples

The following line exempts URLs to the domain xyz.com from having to authenticate using the LDAP server:

```
dest_domain=xyz.com server="auth_bypass"
```

logs.config

The logs.config file establishes and formats custom transaction log files.

For Traffic Server to create the custom log files you define, you must enable the custom logging option by setting the proxy.config.log2.custom_logs_enabled variable to 1, as follows:

```
CONFIG proxy.config.log2.custom_logs_enabled INT 1
```

Refer to *records.config, on page 258* for information about setting records.config variables. If the proxy.config.log2.custom_logs_enabled variable is set to 0 (disabled), any custom log files set in the logs.config file are disabled.

Important After you modify the logs.config file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's bin directory your working directory, then run the traffic_line -x command. In Windows, open a Command Prompt window, cd to the bin directory (located in the Traffic Server installation directory), then run the traffic_line -x command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line in logs.config establishes and formats a custom transaction log file. Lines consist of the following fields, separated by colons (:).

Field	Allowed Inputs
format	All lines must begin with the word format.
activation flag	enabled disabled
unique format identifier	Use a unique integer for each custom log file you create.
format name	Enter a name for the format you define.
format string	Enter a printf-style format string specifying the field symbols to be displayed and how they should look in ASCII. Refer to *Appendix E* for a list of the available field symbols and their meanings. Field symbols are indicated by %<*field_symbol*> format. For example, to indicate that chi is the client host IP and not the string chi to be printed, enter %<chi>.
file name	Enter a name for the custom log file you create.
file type	ASCII BINARY
file header data	none—Enter none if you do not want header text. header text—If you want your custom log file to have a header, enter the appropriate text here.

Examples

The following example shows a custom log file named `minimalist`. It records the following information:

- ✔ The client host IP address (`chi`)
- ✔ The client request universal resource identifier (`cqu`)
- ✔ The proxy response status code (`pssc`)

```
format:enabled:1:minimal:%<chi> / %<cqu> / %<pssc>:minimalist:ASCII:none
```

Example output to the `minimalist` file is as follows:

```
123.12.3.123 / GET http://earth/ocean/index.html HTTP/1.0 / 200
```

The following example shows a log file named `test`. It records the following information:

- ✔ The User-Agent value of the client request header (cqh)
- ✔ The Retry-After value of the proxy response header (psh)

```
format:enabled:1:test:%<{User-Agent}cqh> %<{Retry-After}psh>:test:ASCII:none
```

WELF (WebTrends Enhanced Log Format)

Traffic Server supports WELF, the WebTrends Enhanced Log format, so that you can analyze Traffic Server log files with WebTrends reporting tools. A predefined custom format for WELF is provided in the `logs.config` file. To create a WELF format log file, comment out the following section at the end of the file and replace `<FORMAT_ID>` with a unique integer.

```
#format:enabled:<FORMAT_ID>:welf:id=firewall time="%<cqtd> %<cqtt>" fw=%<phn>
pri=6 proto=%<cqus> duration=%<ttmsf> sent=%<psql> rcvd=%<cqhl> src=%<chi>
dst=%<shi> dstname=%<shn> user=%<caun> op=%<cqhm> arg="%<cqup>" result=%<pssc>
ref="%<{Referer}cqh>" agent="%<{user-agent}cqh>" cache=%<crc>:welf:ASCII:none
#
```

log_hosts.config

To record HTTP/FTP transactions for different origin servers in separate log files, you must list each origin server's hostname in the `log_hosts.config` file.

Note Inktomi recommends that you use the same `log_hosts.config` file on every Traffic Server node in your cluster.

Important After you modify the `log_hosts.config` file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's `bin` directory your working directory, then run the `traffic_line -x` command. In Windows, open a Command Prompt window, `cd` to the `bin` directory (located in the Traffic Server installation directory), then run the `traffic_line -x` command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line in the `log_hosts.config` file has the following format:

```
hostname
```

where *`hostname`* is the hostname of the origin server.

Examples

The following lines in the `log_hosts.config` file configure Traffic Server to create a separate log files containing all HTTP/FTP transactions for the origin servers `webserver1`, `webserver2`, and `webserver3` if the HTTP host log splitting option is enabled.

```
webserver1
webserver2
webserver3
```

For information about enabling the HTTP host log splitting option, refer to *HTTP host log splitting, on page 170*.

logs_xml.config

This is the configuration file for the most configurable form of Traffic Server logging. This file defines log files, their formats, filters, as well as processing options. The format of this file is modeled after XML, the eXtensible Markup Language.

Use this file instead of the `logs.config` file when you need to establish fine-grained control over the format and disposition of custom transaction logs. You can enable or disable this extended format logging by editing the following variable in the `records.config` file:

```
CONFIG proxy.config.log2.xml_logs_config
```

When the `proxy.config.log2.xml_logs_config` variable is set to 1, Traffic Server refers to the XML-based log configuration file specified in `proxy.config.log2.xml_config_file` to determine the extended custom logging specifications. This file, `logs_xml.config`, serves as the default XML-based configuration file.

For information about setting `records.config` variables, refer to *records.config, on page 258*.

Format

You define all custom log files, formats, and filters in a single XML-based configuration file. This file contains specifications of the following types:

- ✔ `LogFormat` specifies the fields to be gathered from each protocol event access.
- ✔ `LogFilter` specifies the filters that are used to include or exclude certain entries being logged based on the value of a field within that entry.
- ✔ `LogObject` specifies an object that contains a particular format, a local file name, filters, and potentially multiple collation servers.

Note The `logs_xml.config` file ignores extra white-space, blank lines, and all comments.

LogFormat

LogFormat specifications can consist of the following tags:

Field	Allowed Inputs
<Name = "valid_format_name"/>	(Required) Valid format names include any name except squid, common, extended, or extended2, which are pre-defined formats. There is no default for this tag.

Field	Allowed Inputs
<Format = "valid_format_specification"/>	(Required) A valid format specification is a printf-style string describing each log entry when formatted for ascii output. Use '%<field>' as placeholders for valid Inktomi field names. For more information, refer to *Inktomi custom logging fields, on page 302*. The specified field can be of two types: ▮ Simple. For example, %<cqu> ▮ A field within a container, such as an HTTP header or an Inktomi stat. Fields of this type have the syntax: '%<{field}container>'.
<Interval = "aggregate_interval_secs"/>	Use this tag when the format contains aggregate operators. The value "aggregate_interval_secs" represents the number of seconds between individual aggregate values being produced. The valid set of aggregate operators are: ▮ COUNT ▮ SUM ▮ AVG ▮ FIRST ▮ LAST

LogFilters

LogFilter specifications can consist of the following tags:

Field	Allowed Inputs
<Name = "valid_filter_name"/>	(Required) All filters must be uniquely named.
<Action = “valid_action_field”/>	(Required) ACCEPT or REJECT. This instructs Traffic Server to either accept or reject records satisfying the condition of the filter.
<Condition = “valid_log_field valid_operator valid_comparison_value”/>	(Required) This field contains the following elements: ▮ valid_log_field: The field that will be compared against the given value. For more information, refer to *Logging format cross reference, on page 304*. ▮ valid_operator_field: Any one of the following: MATCH, CASE_INSENSITIVE_MATCH, CONTAIN, CASE_INSENSITIVE_CONTAIN. MATCH is true if the field and value are identical (case sensitive). CASE_INSENSITIVE_MATCH is similar to MATCH, only case insensitive. CONTAIN is true if the field contains the value (the value is a substring of the field). CASE_INSENSITIVE_CONTAIN is a case insensitive version of CONTAIN. ▮ valid_comparison_value: Any string or integer matching the field type. For integer values, all of the operators are equivalent and mean that the field must be equal to the specified value. Note: There are no negative comparison operators. If you want to specify a negative condition, use the Action field to REJECT the record.

LogObject

LogObject specifications can consist of the following tags:

Field	Allowed Inputs
<Format = "valid_format_name"/>	(Required) Valid format names include the pre-defined logging formats, namely squid, common, extended, and extended2, as well as any previously-defined custom Log formats. There is no default for this tag.
<Filename = "local_file_name"/>	(Required) The file name to which the given log file is written in the local filesystem or in a remote collation server. No local log file will be created if you fail to specify this tag. All file names are relative to the default logging directory. If the name does not contain an extension (for example `squid`), the extension `.log` is automatically appended to it for ASCII logs and `.blog` for binary logs. (See *<Mode = "valid_logging_mode"/>*.) If you do not want an extension to be added, end the filename with a single dot (.). For example, `squid.`
<Mode = "valid_logging_mode"/>	Valid logging modes include ASCII, binary, and ASCII_PIPE. The default is ASCII. Use the ASCII_PIPE mode to write an XML-based custom log file to a pipe so that the logging data is sent to a buffer in memory. Other processes can then read the data using standard I/O functions. The advantage of using this option is that Traffic Server does not have to write to disk, freeing disk space for other tasks. In addition, writing to a pipe does not stop when logging space is exhausted because the pipe does not use disk space. Note: If you are using a collation server, the log is written to a pipe on the collation server. A local pipe is created even before a transaction is processed so that you can see the pipe right after Traffic Server starts. However, pipes on a collation server *are* created when Traffic Server starts.
<CollationHosts = "list_of_valid_hostnames"/>	A comma-separated list of collation servers to which all log entries (for this object) are forwarded. Collation servers can be specified by name or IP address. Specify the collation port with a colon after the name (for example, `host:port`).
<Filters = "list_of_valid_filter_names"/>	A comma-separated list of names of any previously defined log filters. If more than one filter is specified, all filters must accept a record for the record to be logged.
<Protocols = "list_of_valid_protocols"/>	A comma-separated list of the protocols this object should log. Valid protocol names include HTTP, NNTP, ICP.
<ServerHosts = "list_of_valid_servers"/>	A comma-separated list of valid hostnames.This tag indicates that only entries from the named servers will be included in the file.

Examples

The following is an example of a `LogFormat` specification collecting information using three common fields:

```
<LogFormat>
  <Name = "minimal"/>
  <Format = "%<chi> : %<cqu> : %<pssc>"/>
</LogFormat>
```

The following is an example of a `LogFormat` specification using aggregate operators:

```
<LogFormat>
  <Name = "summary"/>
  <Format = "%<LAST(cqts)> : %<COUNT(*)> : %<SUM(psql)>"/>
  <Interval = "10"/>
</LogFormat>
```

The following is an example of a `LogFilter` that will cause only `REFRESH_HIT` entries to be logged:

```
<LogFilter>
  <Name = "only_refresh_hits"/>
  <Action = "ACCEPT"/>
  <Condition = "%<pssc> MATCH REFRESH_HIT"/>
</LogFilter>
```

Note When specifying the field in the filter condition, you can omit the `%<>`. This means that the following filter is equivalent to the example directly above:

```
<LogFilter>
  <Name = "only_refresh_hits"/>
  <Action = "ACCEPT"/>
  <Condition = "pssc MATCH REFRESH_HIT"/>
</LogFilter>
```

The following is an example of a `LogObject` specification that creates a local log file for the minimal format defined earlier. The log file name will be `minimal.log` because this is an ASCII log file (the default).

```
<LogObject>
  <Format = "minimal"/>
  <Filename = "minimal"/>
</LogObject>
```

The following is an example of a `LogObject` specification that includes only HTTP requests served by hosts in the domain `company.com` or by the specific server `server.somewhere.com`. Log entries are sent to port 4000 of the collation host `logs.company.com`, and to port 5000 of the collation host 209.131.52.129:

```
<LogObject>
  <Format = "minimal"/>
  <Filename = "minimal"/>
  <ServerHosts = "company.com,server.somewhere.com"/>
  <Protocols = "http"/>
  <CollationHosts = "logs.company.com:4000,209.131.52.129:5000"/>
</LogObject>
```

WELF (WebTrends Enhanced Log Format)

Traffic Server supports WELF, the WebTrends Enhanced Log format, so that you can analyze Traffic Server log files with WebTrends reporting tools. A predefined `<LogFormat>` that is compatible with WELF is provided at the end of the `logs.config` file (shown below). To create a WELF format log file, create a `<LogObject>` that uses this predefined format.

```
<LogFormat>
  <Name = "welf"/>
  <Format = "id=firewall time=\"%<cqtd> %<cqtt>\" fw=%<phn> pri=6 proto=%<cqus>
duration=%<ttmsf> sent=%<psql> rcvd=%<cqhl> src=%<chi> dst=%<shi> dstname=%<shn>
user=%<caun> op=%<cqhm> arg=\"%<cqup>\" result=%<pssc> ref=\"%<{Referer}cqh>\"
agent=\"%<{user-agent}cqh>\" cache=%<crc>"/>
</LogFormat>
```

mgmt_allow.config

The mgmt_allow.config file specifies the IP addresses of remote hosts allowed to access the Traffic Manager UI. If no entries exist in the mgmt_allow.config file, all remote hosts are allowed to access the Traffic Manager UI.

Important After you modify the mgmt_allow.config file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's bin directory your working directory, then run the traffic_line -x command. In Windows, open a Command Prompt window, cd to the bin directory (located in the Traffic Server installation directory), then run the traffic_line -x command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line in the mgmt_allow.config file has the following format:

```
src_ip=IPaddress_or_range_of_IPaddresses action=ip_allow
```

where *IPaddress_or_range_of_IPaddresses* is the IP address or range of IP addresses allowed to access the Traffic Manager UI.

Examples

The following line in the mgmt_allow.config file allows only one user to access the Traffic Manager UI:

```
src_ip=123.12.3.123 action=ip_allow
```

The following line allows a range of IP addresses to access the Traffic Manager UI:

```
src_ip=123.12.3.000-123.12.3.123 action=ip_allow
```

nntp_access.config

The `nntp_access.config` file controls user access to news articles cached by the Traffic Server. Each line in the `nntp_access.config` file describes the access privileges for a particular group of clients.

Important After you modify the `nntp_access.config` file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's `bin` directory your working directory, then run the `traffic_line -x` command. In Windows, open a Command Prompt window, `cd` to the `bin` directory (located in the Traffic Server installation directory), then run the `traffic_line -x` command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line begins with a specific client group. There are three ways of specifying groups of clients: by IP range, domain, or hostname. For example:

```
ip=0.0.0.0-255.255.255.255
ip=127.0.0.1
domain=inktomi.com
hostname=myhost.mydomain.com
```

Following the client group is an access directive. The access directive is of the form `access=`*`value`*. The allowed access values are `allow`, `deny`, `basic`, `generic`, and `custom`. Depending on the access directive, you can further specify an authenticator program, users, and passwords, as in the following examples:

```
ip=127.0.0.1 access="generic" authenticator="homebrew" user="joe"
hostname=myhost.com access="basic" user="joe" pass="bob"
```

The following table lists the access directive options:

If access is ...	authenticator is ...	user is ...	pass is ...
`allow`	not required	not required	not required
`deny`	not required	not required	not required
`basic`	not required	required	optional
`generic`	optional	not required	not required
`custom`	required	optional; but the only allowed entry is the string 'required' (see example)	optional; but the only allowed entry is the string 'required' (see example)

The following is an example of *custom* access:

```
ip=127.0.0.1 access="custom" authenticator="hb" user=required pass=required
```

nntp_servers.config

The `nntp_servers.config` file configures:

- ✔ The Traffic Server's parent NNTP servers
- ✔ The news groups you want the Traffic Server to observe
- ✔ The type of NNTP activity you want the Traffic Server to do; for example, caching news articles on demand, posting news articles, receiving news feeds
- ✔ The network interface the Traffic Server uses to contact the parent NNTP server

Important After you modify the `nntp_servers.config` file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's `bin` directory your working directory, then run the `traffic_line -x` command. In Windows, open a Command Prompt window, `cd` to the `bin` directory (located in the Traffic Server installation directory), then run the `traffic_line -x` command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line in the `nntp_servers.config` file must have the following format:

```
hostname group-wildmat priority interface
```

The *`hostname`* and *`group-wildmat`* tags are required; *`priority`* and *`interface`* are optional. The following table describes allowed values:

Tag	Description
`hostname`	Enter one of the following: hostname hostname:port IP address IP address:port .block
`group-wildmat`	Enter a comma-separated list of group names and *list files* in *wildmat* format (use * as a wildcard). The list files options are the following: `subscriptions`, `distributions`, and `distrib.pats`. Make sure there are no spaces in the list. Use the prefix ! to indicate groups that are *not* included in the list. The list is processed in reverse order, so more specific restrictions should be placed later in the list. Examples: ▪ `*,!distrib.pats` Do not include any `distrib.pats` files, but do include all others. ▪ `*,!alt.*` Do not include any groups of the form `alt.*`, but do include all others. ▪ `talk.religion.*,!talk.religion.barney,subscriptions` Include only subscriptions from all `talk.religion.*` groups, excluding `talk.religion.barney`.
`priority`	This tag tells the Traffic Server how to treat the specified host and news groups. Use one of the following options:

Tag	Description
	■ <no priority tag> If you do not use a priority tag, Traffic Server caches articles from the specified news groups on demand. If you specify multiple groups (such as `alt.*`), the Traffic Server will maintain a group list and will poll the parent NNTP server regularly to check for changes in the group list.
	■ feed The Traffic Server will receive news feeds for the specified groups as the parent NNTP server receives news feeds. The Traffic Server will not cache articles on demand, since it will simply have them. **Caution**: If Traffic Server is clustered, make sure that your news server sends feeds to *one* of the nodes in the cluster to avoid possible article numbering conflicts. **Note**: A "feed" line in `nntp_servers.config` must be preceded by a "cache on demand" line. The Traffic Server needs to be aware of the news server and its groups before it can receive news feeds. See the examples following this table.
	■ push The Traffic Server can both receive news feeds and cache articles on demand. **Caution**: If Traffic Server is clustered, make sure that your news server sends feeds to *one* of the nodes in the cluster, to avoid possible article numbering conflicts.
	■ pull The Traffic Server actively pulls (caches) all articles from these news groups at a frequency you specify in the Traffic Server Manager UI. The Traffic Server does not wait for user requests. **Note**: A "pull" line in `nntp_servers.config` must be preceded by a "cache on demand" line. The reason is that Traffic Server needs to be aware of the news server and its groups before it can pull articles from a specific group. See the examples following this table.
	■ pullover The Traffic Server actively pulls the overview database for the news groups, but retrieves news articles on demand. **Note**: A "pullover" line in `nntp_servers.config` must be preceded by a "cache on demand" line. The Traffic Server needs to be aware of the news server and its groups before it can pull overviews from a specific group. See the examples following this table.
	■ dynamic The Traffic Server automatically decides, based on usage patterns, whether a group should be "pull", "pullover", or demand retrieval-based.
	■ Enter a positive integer The Traffic Server retrieves articles on demand from the specified server according to the assigned priority. The default priority is 0. Multiple servers assigned the same priority are accessed in a round-robin fashion.
	■ post Articles to be posted to the specified news groups are sent to the specified server.
`interface`	Enter the network interface the Traffic Server uses to contact the parent NNTP server.

Examples

The following example configures Traffic Server to block all requests from `rec.*` groups, except for `rec.soccer`:

```
.block rec.*,!rec.soccer,
```

The following examples define pull and pullover groups.

```
comp.qqqq.com:9999 comp.* feed 10.3.3.2
news.qqqq.com * 2
```

Note Every line designating a pull or pullover group must be preceded by a *cache on demand* line as follows:

```
comp.webhost.com alt.*
comp.webhost.com alt.bicycles pull
```

```
news.inktomi.com ink.*
news.inktomi.com ink.* pullover
```

parent.config

The parent.config file identifies the HTTP parent proxies used in an HTTP cache hierarchy. This file allows you to:

✔ Set up parent cache hierarchies, with multiple parents and parent failover

✔ Configure selected URL requests to bypass parent proxies

For the parent.config file to take effect, HTTP parent caching option must be enabled in the Traffic Manager UI. Refer to *Enabling the HTTP parent caching option, on page 93*.

Important After you modify the parent.config file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's bin directory your working directory, then run the traffic_line -x command. In Windows, open a Command Prompt window, cd to the bin directory (located in the Traffic Server installation directory), then run the traffic_line -x command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line in the parent.config file must contain a parent caching rule. Traffic Server recognizes three space-delimited tags:

```
primary destination=value secondary specifier=value action=value
```

The following table lists the possible primary destinations and their allowed values:

Primary Destination	Allowed Value
dest_domain	Requested domain name
dest_host	Requested hostname
dest_ip	Requested IP address
url_regex	Regular expression to be found in a URL

The secondary specifiers are optional in the parent.config file. The following table lists the possible secondary specifiers and their allowed values:

Secondary Specifiers	Allowed Value
time	A time range, such as 08:00-14:00 during which the parent cache is used to serve requests.
src_ip	The IP address of the client
prefix	A prefix in the path part of a URL
suffix	A file suffix in the URL
port	A requested URL port
method	A request URL method; one of the following: get post put trace
scheme	A request URL protocol; one of the following: HTTP FTP

The following tables lists the possible actions and their allowed values:

Action	Allowed Value
`parent`	Enter an ordered list of parent proxies. If the request cannot be handled by the last parent server in the list, it will be routed to the origin server.
`round_robin`	▮ true Enter true if you want the Traffic Server to go through the parent cache list in a round robin. ▮ strict Enter strict if you want Traffic Server machines to serve requests strictly in turn. For example, machine ts-sun8 serves the first request, ts-sun9 serves the second request, and so on. ▮ false Enter false if you do not want round robin selection to occur.
`go_direct`	▮ true Enter true if you want requests to bypass parent hierarchies and go directly to the origin server. ▮ false Enter false if you do not want requests to bypass parent hierarchies.

Examples

The following rule sets up a parent cache hierarchy consisting of the Traffic Server (which is the child) and two parents, `p1` and `p2`. All `get` requests, if they cannot be served by the Traffic Server, are routed to the first parent server, `p1.x.com`. If they are not in the first parent server, they are routed to the second parent server, `p2.y.com`. Because `round_robin=true`, the parent servers are queried in a round robin fashion.

```
dest_domain=. method=get parent="p1.x.com:8080; p2.y.com:8080" round_robin=true
```

The following rule tells the Traffic Server to route all requests containing the regular expression `politics` and the path /`viewpoint` directly to the origin server (bypassing any parent hierarchies).

```
url_regex=politics prefix=/viewpoint go_direct=true
```

Every line in the `parent.config` file must contain either a `parent=` or `go_direct=` directive.

A parent cache entry in the **Parent Caching** section of the **Routing** page, described on *page 94*, is entered as the bottom line in the `parent.config` file (meaning that this entry serves as a default). For example, if you enter `parent1:8080` in the `Parent Cache` field on the **Routing** page, the following line is entered at the bottom of the `parent.config` file:

```
dest_domain=. parent=parent1:8080
```

partition.config

The `partition.config` file lets you manage your cache space more efficiently and restrict disk usage by creating cache partitions of different sizes for specific protocols. You can further configure these partitions to store data from certain origin servers and/or domains in the `hosting.config` file (refer to *hosting.config, on page 237*).

For step by step instructions on partitioning the cache, refer to *Partitioning the cache, on page 104*.

Important The partition configuration must be the same on all nodes in a cluster.

You must stop Traffic Server before you change the cache partition size and protocol assignment.

Format

For each partition you want to create, enter a line with the following format:

```
partition=partition_number  scheme=protocol_type  size=partition_size
```

where:

`partition_number` is a number between 1 and 255 (the maximum nuber of partitions is 255).

`protocol_type` is either `http` or `mixt` (all streaming media content is stored in the `mixt` partition, all other content is stored in the `http` partition).

`partition_size` is the amount of cache space allocated to the partition. This value can be either a percentage of the total cache space or an absolute value. The absolute value must be a multiple of 128 MB, where 128 MB is the smallest value. If you specify a percentage, the size is rounded down to the closest multiple of 128 MB. Each partition is striped across several disks to achieve parallel I/O. For example, if there are 4 disks, a 1 GB partition will have 256 MB on each disk (assuming each disk has enough free space available).

Note If you do not allocate all the disk space in the cache, the extra disk space is not used. You can use the extra space at a later time to create new partitions without deleting and clearing the existing partitions.

Examples

The following example partitions the cache evenly between HTTP and streaming media requests:

```
partition=1 scheme=http size=50%
partition=2 scheme=mixt size=50%
```

records.config

The records.config file is a list of configurable variables that Traffic Server software uses. This section lists and describes these variables.

Many of the variables in the records.config file are set automatically when you set configuration options in Traffic Manager or Traffic Line. Certain configuration options can be set only by editing variables manually in the records.config file.

Warning Do not change the records.config variables unless you are certain of the effect. Many variables are coupled, meaning that they interact with other variables. Changing a single variable in isolation could cause the Traffic Server to fail. Whenever possible, use Traffic Manager or Traffic Line to configure the Traffic Server.

Important After you modify the records.config file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's bin directory your working directory, then run the traffic_line -x command. In Windows, open a Command Prompt window, cd to the bin directory (located in the Traffic Server installation directory), then run the traffic_line -x command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each variable has the following format:

```
CONFIG variable_name DATATYPE variable_value
```

where *DATATYPE* is INT (an integer), STRING (a string), or FLOAT (a floating point).

Examples

In the following example, the variable proxy.config.proxy_name is of datatype string and its value is my_server. This means that the name of the Traffic Server proxy is my_server.

```
CONFIG proxy.config.proxy_name STRING my_server
```

In the following example, the variable proxy.config.arm.enabled is a yes/no flag. A value of 0 (zero) disables the option. A value of 1 enables the option:

```
CONFIG proxy.config.arm.enabled INT 0
```

In the following example, the variable sets the cluster startup timeout to 10 seconds.

```
CONFIG proxy.config.cluster.startup_timeout INT 10
```

Configuration variables

The following table describes the configuration variables listed in the `records.config` file.

Configuration Variable **Data Type**	**Default Value**	**Description**
	System Variables	
proxy.config.proxy_name STRING		Specifies the name of the Traffic Server node.
proxy.config.bin_path STRING	bin	Specifies the location of Traffic Server's bin directory.
proxy.config.proxy_binary STRING	traffic_server (UNIX) traffic_server.exe (Windows)	Specifies the name of the executable that runs the traffic_server process.
proxy.config.proxy_binary_opts STRING	-M	Specifies the command-line options for starting Traffic Server.
proxy.config.manager_binary STRING	traffic_manager (UNIX) traffic_manager.exe (Windows)	Specifies the name of the executable that runs the traffic_manager process.
proxy.config.cli_binary STRING	traffic_line (UNIX) traffic_line.exe (Windows)	Specifies the name of the executable that runs the command-line interface (Traffic Line).
proxy.config.watch_script STRING	traffic_cop	Specifies the name of the executable that runs the traffic_cop process.
proxy.config.env_prep STRING	example_prep.sh (UNIX) example_prep.bat (Windows)	Specifies the script that is executed before the traffic_manager process spawns the traffic_server process.
proxy.config.config_dir STRING	config	Specifies the directory that contains the Traffic Server configuration files.
proxy.config.temp_dir STRING	/tmp	Specifies the directory used for Traffic Server temporary files.
proxy.config.syslog_facility STRING	LOG_DAEMON	Specifies the facility used to record system log files. (UNIX only.) Refer to *Understanding Traffic Server log files, on page 154*.
proxy.config.cop.core_signal INT	0	Specifies the signal that is sent to traffic_cop's managed processes to stop them. Unix only. 0 = no signal is sent.

Configuration Variable Data Type	Default Value	Description
	Local Manager	
proxy.config.lm.sem_id INT	11452	Specifies the semaphore ID for the local manager.
proxy.config.cluster.type INT	3	Sets the clustering mode: 1 = full-clustering mode 2 = management-only mode 3 = no clustering
proxy.config.cluster.rsport INT	8088	Specifies the reliable service port. The reliable service port is used to send configuration information between the nodes in a cluster. All nodes in a cluster must use the same reliable service port.
proxy.config.cluster.mcport INT	8089	Specifies the multicast port. The multicast port is used for node identification. All nodes in a cluster must use the same multicast port.
proxy.config.cluster.mc_group_addr STRING		Specifies the multicast address for cluster communications. All nodes in a cluster must use the same multicast address.
proxy.config.cluster.mc_ttl INT	1	Specifies the multicast Time to Live for cluster communications.
proxy.config.cluster.log_bogus_mc_msgs INT	1	Enables (1) or disables (0) logging of bogus multicast messages.
proxy.config.admin.html_doc_root STRING	ui	Specifies the document root for the Traffic Manager UI.
proxy.config.admin.web_interface_port INT	8081	Specifies the Traffic Manager port.
proxy.config.admin.autoconf_port INT	8083	Specifies the auto-configuration port.
proxy.config.admin.overseer_port INT	8082	Specifies the port used for retrieving and setting statistics and configuration variables.
proxy.config.admin.admin_user STRING	admin	Specifies the administrator ID that controls access to the Traffic Manager UI.
proxy.config.admin.admin_password STRING		Specifies the encrypted administrator password that controls access to the Traffic Manager UI. You cannot edit the password, however, you can specify a value of `NULL` to clear the password. Refer to *If you forget the administrator password, on page 137*.
proxy.config.admin.basic_auth INT	1	Enables (1) or disables (0) basic user authentication to control access to the Traffic Manager UI. Note: If basic authentication is *not* enabled, any user can access the Traffic Manager to monitor and configure Traffic Server.

Configuration Variable Data Type	Default Value	Description
proxy.config.admin.use_ssl INT	0	Enables the Traffic Manager SSL option for secure communication between a remote host and the Traffic Manager.
proxy.config.admin.ssl_cert_file STRING	private_key.pem	Specifies the file name of the SSL certificate installed on the Traffic Server system for secure communication between a remote host and the Traffic Manager UI.
proxy.config.admin.number_config_bak INT	3	Specifies the maximum number of copies of rolled configuration files to keep.
proxy.config.admin.user_id STRING	inktomi	Specifies the non-privileged user account designated to Traffic Server. (UNIX only.)
proxy.config.admin.ui_refresh_rate INT	30	Specifies the refresh rate for the display of statistics in the Monitor pages of the Traffic Manager UI.
proxy.config.admin.log_mgmt_access INT	0	Enables (1) or disables (0) logging of all Traffic Manager UI transactions to the lm.log file.
proxy.config.admin.log_resolve_hostname INT	1	When enabled (1), the hostname of the client connecting to Traffic Manager is recorded in the lm.log file. When disabled (0), the IP address of the client connecting to Traffic Manager is not recorded in the lm.log file.
	Process Manager	
proxy.config.process_manager.mgmt_port INT	8084	Specifies the port used for internal communication between the traffic_manager process and the traffic_server process.
	Virtual IP Manager	
proxy.config.vmap.enabled INT	0	Enables (1) or disables (0) the Virtual IP option. Refer to *Virtual IP failover, on page 84*.
proxy.config.vmap.mode INT	0	Refer to the *Media-IXT User's Guide*.
	Alarm Configuration	
proxy.config.alarm.bin STRING	example_alarm_bin.sh (UNIX) example_alarm.bat (Windows)	Specifies the name of the script file that sends E-mail to alert someone of Traffic Server problems. The default file is a sample script. You must edit the script to suit your needs. Refer to *Configuring Traffic Server to E-mail alarms, on page 116*.

Configuration Variable Data Type	Default Value	Description
proxy.config.alarm.abs_path STRING	NULL	Specifies the full path to the script file that sends E-mail to alert someone of Traffic Server problems.
	Inktomi PhoneHome	
proxy.config.phone_home.phone_home_send_info_enabled INT	1	Enables (1) or disables (0) the PhoneHome data sending option. When enabled, Traffic Server sends statistics and configuration information back to a central Inktomi server. If this variable is disabled (0), and the variable proxy.config.phone_home.phone_home_frequency is set to a value other than -1, Traffic Server contacts the central Inktomi server to indicate that it is up and running but does *not* send statistics and configuration information.
proxy.config.phone_home.phone_home_data_encryption_enabled INT	1	Enables (1) or disables (0) data encryption for sending statistics and configuration information back to a central Inktomi server**.**
proxy.config.phone_home.phone_home_frequency INT	86400	Specifies how often (in seconds) Traffic Server contacts the central Inktomi server. The default value of 86400 seconds contacts the Inktomi server once a day. A value of -1 disables the PhoneHome option. Traffic Server does not contact the central Inktomi server.
proxy.config.phone_home.phone_home_server STRING	sm.inktomi.com:80	Specifies the name of the central Inktomi server that Traffic Server contacts to send statistics and configuration information. Note: You must include the port number in the server name.
proxy.config.phone_home.phone_home_path STRING	/cgi-bin/phone_home.cgi	Specifies the location of the script on the central Inktomi server to which Traffic Server sends statistics and configuration information.
proxy.config.phone_home.phone_home_id STRING		Specifies the unique ID assigned to the Traffic Server. The central Inktomi server uses this ID when collecting statistics and configuration information from the Traffic Server.
	ARM (Transparency Configuration)	
proxy.config.arm.enabled INT	0	Enables (1) or disables (0) the Traffic Server transparency option for transparent proxy caching. Refer to *Chapter 4, Transparent Proxy Caching.*
proxy.config.arm.ignore_ifp INT	0	Configures Traffic Server to ignore the interface when sending packets back to the client if NAT rules are applied.
proxy.config.arm.nat_config_file STRING	NULL	Specifies the file name of the NAT configuration file (`ipnat.config`).

Configuration Variable **Data Type**	**Default Value**	**Description**
proxy.config.arm.always_query_dest INT	0	Configures Traffic Server to ignore host headers and always ask for the IP address of origin servers.
proxy.config.arm.acl_filename_master STRING	NULL	Specifies the name of the master bypass configuration file (`bypass.config`).
proxy.config.arm.bypass_dynamic_enabled INT	0	Enables (1) or disables (0) the adaptive bypass option to bypass the proxy and go directly to the origin server when clients or servers cause problems.
proxy.config.arm.bypass_use_and_rules_bad_client_request INT	0	Enables (1) or disables (0) dynamic source/destination bypass in the event of non-HTTP traffic on port 80. Note: The variable proxy.config.arm.bypass_on_bad_client_request must also be enabled for this option to work.
proxy.config.arm.bypass_use_and_rules_400 INT	0	Enables (1) or disables (0) dynamic generation of source/destination bypass rules when an origin server returns a 400 error. Note: The variable proxy.config.arm.bypass_on_400 must also be enabled for this option to work.
proxy.config.arm.bypass_use_and_rules_401 INT	0	Enables (1) or disables (0) dynamic generation of source/destination bypass rules when an origin server returns a 401 error. Note: The variable proxy.config.arm.bypass_on_401 must also be enabled for this option to work.
proxy.config.arm.bypass_use_and_rules_403 INT	0	Enables (1) or disables (0) dynamic generation of source/destination bypass rules when an origin server returns a 403 error. Note: The variable proxy.config.arm.bypass_on_403 must also be enabled for this option to work.
proxy.config.arm.bypass_use_and_rules_405 INT	0	Enables (1) or disables (0) dynamic generation of source/destination bypass rules when an origin server returns a 405 error. Note: The variable proxy.config.arm.bypass_on_405 must also be enabled for this option to work.
proxy.config.arm.bypass_use_and_rules_406 INT	0	Enables (1) or disables (0) dynamic generation of source/destination bypass rules when an origin server returns a 406 error. Note: The variable proxy.config.arm.bypass_on_406 must also be enabled for this option to work.
proxy.config.arm.bypass_use_and_rules_408 INT	0	Enables (1) or disables (0) dynamic generation of source/destination bypass rules when an origin server returns a 408 error. Note: The variable proxy.config.arm.bypass_on_408 must also be enabled for this option to work.

Configuration Variable Data Type	Default Value	Description
proxy.config.arm.bypass_use_and_rules_500 INT	0	Enables (1) or disables (0) dynamic generation of source/destination bypass rules when an origin server returns a 500 error. Note: The variable proxy.config.arm.bypass_on_500 must also be enabled for this option to work.
proxy.config.arm.bypass_on_bad_client_request INT	0	Enables (1) or disables (0) dynamic destination bypass in the event of non-HTTP traffic on port 80.
proxy.config.arm.bypass_on_400 INT	0	Enables (1) or disables (0) dynamic generation of destination bypass rules when an origin server returns a 400 error.
proxy.config.arm.bypass_on_401 INT	0	Enables (1) or disables (0) dynamic generation of destination bypass rules when an origin server returns a 401 error.
proxy.config.arm.bypass_on_403 INT	0	Enables (1) or disables (0) dynamic generation of destination bypass rules when an origin server returns a 403 error.
proxy.config.arm.bypass_on_405 INT	0	Enables (1) or disables (0) dynamic generation of destination bypass rules when an origin server returns a 405 error.
proxy.config.arm.bypass_on_406 INT	0	Enables (1) or disables (0) dynamic generation of destination bypass rules when an origin server returns a 406 error.
proxy.config.arm.bypass_on_408 INT	0	Enables (1) or disables (0) dynamic generation of destination bypass rules when an origin server returns a 408 error.
proxy.config.arm.bypass_on_500 INT	0	Enables (1) or disables (0) dynamic generation of destination bypass rules when an origin server returns a 500 error.
proxy.config.header.parse.no_host_url_redirect STRING	NULL	Specifies the URL to which to redirect requests with no host headers (reverse proxy).
(ARM) Load Shedding Configuration		
proxy.config.arm.loadshedding.max_connections INT	1000000	Specifies the maximum number of client connections allowed before Traffic Server starts forwarding incoming requests directly to the origin server.
proxy.config.arm.loadshedding.min_percent_bypass INT	0	Specifies the minimum percent of requests that can bypass the Traffic Server cache. 0 (zero) disables load shedding
proxy.config.arm.loadshedding.max_percent_bypass INT	0	Specifies the maximum percent of requests that can bypass the Traffic Server cache. 0 (zero) disables load shedding Important: Do not set this variable to above 99.

Configuration Variable Data Type	Default Value	Description
proxy.config.arm.loadshedding.max_latency INT	10000	Specifies the maximum amount of time allowed in milliseconds between when a request is received and when it is served. If this value is exceeded, load shedding takes effect.
	LDAP	
proxy.config.ldap.auth.enabled INT	0	Enables (1) or disables (0) LDAP-based *basic* proxy authentication.
proxy.config.ldap.cache.size INT	5000	Specifies the maximum number of entries allowed in the LDAP cache. When modifying this value, update the value of proxy.config.ldap.cache.storage_size proportionally. For example, if you double the cache size, also double the cache storage size.
proxy.config.ldap.cache.storage_size INT	24582912	Specifies the size of the LDAP cache in bytes. This is directly related to the number of entries in the cache. When modifying this value, update the value of proxy.config.ldap.cache.size proportionally. For example, if you double the storage size, also double the cache size. Accidentally modifying this variable randomly will cause the LDAP subsystem to stop functioning.
proxy.config.ldap.auth.ttl_value INT	3000	Specifies the amount of time (in minutes) that entries in the cache will remain valid.
proxy.config.ldap.auth.purge_cache_on_auth_fail INT	0	When enabled (1), configures Traffic Server to delete the authorization entry for the client in the LDAP cache if authorization fails.
proxy.config.ldap.auth.multiple.ldap_servers.enabled INT	0	Enables (1) or disables (0) the use of multiple LDAP servers for LDAP-based authentication. You must enable this feature when enabling LDAP authentication bypass (proxy.config.ldap.auth.bypass. enabled).
proxy.config.ldap.auth.bypass.enabled INT	0	Enables (1) or disables (0) client access to specific URLs without being authenticated by an LDAP server. When enabling this feature, you must also enable the variable proxy.config.ldap.auth.multiple ldap_servers.enabled. You specify the URLs in the multiple LDAP server configuration file (ldap_servers.config file).
proxy.config.ldap.proc.ldap.server.name STRING	NULL	Specifies the LDAP server name.
proxy.config.ldap.proc.ldap.server.port INT	0	Specifies the LDAP port.

Configuration Variable Data Type	Default Value	Description
proxy.config.ldap.proc.ldap.base.dn STRING	NULL	Specifies the LDAP base Distinguished Name (DN). Obtain this value from your LDAP administrator.
	HTTP Engine	
proxy.config.http.server_port INT	8080	Specifies the port that Traffic Server uses when acting as a web proxy server for web traffic or when serving web traffic transparently.

proxy.config.http.server_port_attr
STRING
	X	Specifies the server port options. You can specify one of the following:
		C = SERVER_PORT_COMPRESSED
		X = SERVER_PORT_DEFAULT
		T = SERVER_PORT_BLIND_TUNNEL

proxy.config.http.server_other_ports STRING	NULL	Specifies the ports other than the port specified by the variable proxy.config.http.server_port to bind for incoming http requests.
proxy.config.http.ssl_ports STRING	443 563	Specifies the range of ports used for tunneling. Traffic Server allows tunnels only to the specified ports. For example, to retrieve an object using HTTPS via Traffic Server requires establishing a tunnel via Traffic Server to an origin server.
proxy.config.http.insert_request_via_str INT	1	You can specify one of the following: 0 = no extra information is added to the string. 1 = all extra information is added. 2 = some extra information is added.
proxy.config.http.insert_response_via_str INT	1	You can specify one of the following: 0 = no extra information is added to the string. 1 = all extra information is added. 2 = some extra information is added.
proxy.config.http.enable_url_expandomatic INT	1	Enables (1) or disables (0) .com domain expansion, which configures the Traffic Server to attempt to resolve unqualified hostnames by redirecting them to the expanded address, prepended with www. and appended with .com. For example, if a client makes a request to inktomi, Traffic Server redirects the request to www.inktomi.com.
proxy.config.http.no_dns_just_forward_to_parent INT	0	When enabled (1), and if HTTP parent caching is enabled, Traffic Server does no DNS lookups on request hostnames.
proxy.config.http.keep_alive_enabled INT	1	Enables (1) or disables (0) the use of keep alive connections to either origin servers or clients.

Configuration Variable Data Type	Default Value	Description
proxy.config.http.send_http11_requests INT	3	Configures Traffic Server to use HTTP version 1.1 when communicating with origin servers. You can specify one of the following values: 1 = Traffic Server always uses HTTP 1.1 when communicating with origin servers. 2 = Traffic Server uses HTTP 1.1 if the origin server has previously used HTTP 1.1. 3 = Traffic Server uses HTTP 1.1 if the client request is HTTP 1.1 and the origin server has previously used HTTP 1.1. Note: The HTTP version used affects whether a keep-alive connection may be used. If HTTP 1.1 is used, then Traffic Server can use keep-alive connections with pipelining to origin servers. If HTTP 0.9 is used, then Traffic Server does not use keep-alive connections to origin servers. If HTTP 1.0 is used, then a Traffic Server can use keep-alive connections without pipelining to origin servers.
proxy.config.http.origin_server_pipeline INT	1	Configures Traffic Server to use keep-alive connections with or without pipelining when connecting to the origin server. You can specify one of the following values: 0 = Traffic Server does not use keep-alive connections to origin servers. 1 = Traffic Server uses keep-alive connections to origin servers without pipelining. >1 = Traffic Server uses keep-alive connections with pipelining to origin servers.
proxy.config.http.user_agent_pipeline INT	4	Configures Traffic Server to use keep-alive connections with or without pipelining when connecting to the client. You can specify one of the following values: 0 = Traffic Server does not use keep-alive connections to clients. 1 = Traffic Server uses keep-alive connections to clients without pipelining. >1 = Traffic Server uses keep-alive connections with pipelining to clients.
proxy.config.http.share_server_sessions INT	1	Enables (1) or disables (0) the re-use of server sessions.
proxy.config.http.ftp_enabled INT	1	Enables (1) or disables (0) Traffic Server from serving FTP requests sent via HTTP.
proxy.config.http.record_heartbeat INT	0	Enables (1) or disables (0) traffic_cop heartbeat logging.

Configuration Variable Data Type	Default Value	Description
	parent proxy configuration	
proxy.config.http.parent_proxy_routing_enable INT	0	Enables (1) or disables (0) the HTTP parent caching option. Refer to *Chapter 7, Hierarchical Caching*.
proxy.config.http.parent_proxies STRING	NULL	Specifies the parent cache.
proxy.config.http.parent_proxy.retry_time INT	300	Specifies the amount of time allowed between connection retries to a parent cache that is unavailable.
proxy.config.http.parent_proxy.fail_threshold INT	10	Specifies the number of times the connection to the parent cache can fail before Traffic Server considers the parent unavailable.
proxy.config.http.parent_proxy.total_connect_attempts INT	4	Specifies the total number of connection attempts allowed to a parent cache before Traffic Server bypasses the parent or fails the request (depending on the go_direct option in the bypass.config file).
proxy.config.http.parent_proxy.per_parent_connect_attempts INT	2	Specifies the total number of connection attempts allowed per parent if multiple parents are used.
proxy.config.http.parent_proxy.connect_attempts_timeout INT	30	Specifies the timeout value in seconds for parent cache connection attempts.
proxy.config.http.forward.proxy_auth_to_parent INT	0	Configures Traffic Server to send proxy authentication headers on to the parent cache.
	HTTP connection timeouts (secs)	
proxy.config.http.keep_alive_no_activity_timeout_in INT	10	Specifies how long Traffic Server keeps connections to clients open for a subsequent request after a transaction ends.
proxy.config.http.keep_alive_no_activity_timeout_out INT	10	Specifies how long Traffic Server keeps connections to origin servers open for a subsequent transfer of data after a transaction ends.
proxy.config.http.transaction_no_activity_timeout_in INT	120	Specifies how long the Traffic Server should keep connections to clients open if a transaction stalls.
proxy.config.http.transaction_no_activity_timeout_out INT	120	Specifies how long the Traffic Server keeps connections to origin servers open if the transaction stalls.
proxy.config.http.transaction_active_timeout_in INT	7200	Specifies the maximum amount of time Traffic Server can remain connected to a client. If the transfer to the client is not complete before this timeout expires, Traffic Server closes the connection.

Configuration Variable Data Type	Default Value	Description
proxy.config.http.transaction_active_timeout_out INT	7200	Specifies the maximum amount of time Traffic Server waits for fulfillment of a connection request to an origin server. If the Traffic Server does not complete the transfer to the origin server before this timeout expires, the Traffic Server terminates the connection request.
proxy.config.http.accept_no_activity_timeout INT	120	Specifies the timeout interval in seconds before Traffic Server closes a connection that has no activity.
proxy.config.http.background_fill_active_timeout INT	60	Specifies how long Traffic Server continues a background fill before giving up and dropping the origin server connection.
proxy.config.http.background_fill_completed_threshold FLOAT	0.5	Specifies the proportion of total document size already transferred when a client aborts at which the proxy continues fetching the document from the origin server in order to get it into the cache (a *background fill*).
origin server connect attempts		
proxy.config.http.connect_attempts_max_retries INT	6	Specifies the maximum number of connection retries Traffic Server can make when the origin server is not responding.
proxy.config.http.connect_attempts_max_retries_dead_server INT	2	Specifies the maximum number of connection retries Traffic Server can make when the origin server is unavailable.
proxy.config.http.connect_attempts_rr_retries INT	2	Specifies the maximum number of failed connection attempts allowed before a round robin entry is marked as down if a server has round robin DNS entries.
proxy.config.http.connect_attempts_timeout INT	30	Specifies the timeout value in seconds for an origin server connection.
proxy.config.http.down_server.cache_time INT	900	Specifies how long in seconds Traffic Server remembers that an origin server was unreachable.
proxy.config.http.down_server.abort_threshold INT	10	Specifies the number of seconds before Traffic Server marks an origin server as unavailable when a client abandons a request because the origin server was too slow in sending the response header.
proxy users variables		
proxy.config.http.anonymize_remove_from INT	0	When enabled (1), Traffic Server removes the `From` header that accompanies transactions to protect the privacy of your users.
proxy.config.http.anonymize_remove_referer INT	0	When enabled (1), Traffic Server removes the `Referer` header that accompanies transactions to protect the privacy of your site and users.

Configuration Variable Data Type	Default Value	Description
proxy.config.http.anonymize_remove_user_agent INT	0	When enabled (1), Traffic Server removes the `User-agent` header that accompanies transactions to protect the privacy of your site and users.
proxy.config.http.anonymize_remove_cookie INT	0	When enabled (1), Traffic Server removes the `Cookie` header that accompanies transactions to protect the privacy of your site and users.
proxy.config.http.anonymize_remove_client_ip INT	0	When enabled (1), Traffic Server removes `Client-IP` headers for more privacy.
proxy.config.http.anonymize_insert_client_ip INT	0	When enabled (1), Traffic Server inserts `Client-IP` headers to retain the client's IP address.
proxy.config.http.append_xforwards_header INT	0	When enabled (1), Traffic Server appends `X-Forwards` headers to outgoing requests.
proxy.config.http.anonymize_other_header_list STRING	NULL	Specifies the headers that Traffic Server will remove from outgoing requests.
proxy.config.http.snarf_username_from_authorization INT	0	When enabled (1), Traffic Server takes the username and password from the authorization header for LDAP if the authorization scheme is *Basic*.
proxy.config.http.insert_squid_x_forwarded_for INT	0	When enabled (1), Traffic Server adds the client IP address to the `X-Forwarded-For` header.
	security	
proxy.config.http.push_method_enabled INT	0	Enables (1) or disables (0) the HTTP PUSH option that allows you to deliver content directly to the cache without user request. Important: If you enable this option, you must also specify a filtering rule in the filter.config file to allow only certain machines to push content into the cache. Refer to *filter.config, on page 234*.
	cache control	
proxy.config.http.cache.http INT	1	Enables (1) or disables (0) caching of HTTP requests.
proxy.config.http.cache.ftp INT	1	Enables (1) or disables (0) caching of FTP requests sent via HTTP.
proxy.config.http.cache.ignore_client_no_cache INT	0	When enabled (1), Traffic Server ignores client requests to bypass the cache.
proxy.config.http.cache.ims_on_client_no_cache INT	0	When enabled (1), Traffic Server issues a conditional request to the origin server if an incoming request has a `no-cache` header.
proxy.config.http.cache.ignore_server_no_cache INT	0	When enabled (1), Traffic Server ignores origin server requests to bypass the cache.

Configuration Variable Data Type	Default Value	Description
proxy.config.http.cache.cache_responses_to_cookies INT	3	Specifies how cookies are cached. 0 = do not cache any responses to cookies 1 = cache for any content-type 2 = cache only for image types 3 = cache for all but text content-types
proxy.config.http.cache.ignore_authentication INT	0	When enabled (1), Traffic Server ignores `WWW-Authentication` headers in responses. `WWW-Authentication` headers are removed and not cached.
proxy.config.http.cache.cache_urls_that_look_dynamic INT	0	Enables (1) or disables (0) caching of URLs that look dynamic.
proxy.config.http.cache.enable_default_vary_headers INT	1	Enables (1) or disables (0) caching of alternate versions of HTTP objects.
proxy.config.http.cache.when_to_revalidate INT	0	Specifies when to revalidate content. 0 = Use cache directives or heuristic. 1 = Stale if heuristic. 2 = Always stale (always revalidate). 3 = Never stale.
proxy.config.http.cache.when_to_add_no_cache_to_msie_requests INT	0	Specifies when to add `no-cache` directives to Microsoft Internet Explorer requests. You can specify the following: 0 = `no-cache` not added to MSIE requests. 1 = `no-cache` added to IMS MSIE requests. 2 = `no-cache` added to all MSIE requests.
proxy.config.http.cache.required_headers INT	0	Specifies the type of headers required in a request for the request to be cacheable. 0 = no required headers to make document cacheable. 1 = at least `Last-Modified` header required. 2 = explicit lifetime required, `Expires` or `Cache-Control`.
proxy.config.http.cache.max_stale_age INT	604800	Specifies the maximum age allowed for a stale response before it cannot be cached.
proxy.config.http.cache.add_content_length INT	0	When enabled (1), Traffic Server adds the content length header in a request if it is absent.
proxy.config.http.cache.range.lookup INT	1	When enabled (1) Traffic Server looks up range requests in the cache.
	heuristic expiration	
proxy.config.http.cache.heuristic_min_lifetime INT	3600	Specifies the minimum amount of time that a document in the cache can be considered fresh.
proxy.config.http.cache.heuristic_max_lifetime INT	86400	Specifies the maximum amount of time that a document in the cache can be considered fresh.
proxy.config.http.cache.heuristic_lm_factor FLOAT	0.10	Specifies the aging factor for freshness computations.

Configuration Variable Data Type	Default Value	Description
proxy.config.http.cache.fuzz.time INT	240	Specifies the interval in seconds before the document stale time that Traffic Server checks for an early refresh.
proxy.config.http.cache.fuzz.probability FLOAT	0.005	Specifies the probability that a refresh is made on a document during the specified fuzz time.
dynamic content & content negotiation		
proxy.config.http.cache.vary_default_text STRING	Cookie	Specifies the header on which Traffic Server varies for text documents. For example, if you specify `user-agent`, Traffic Server caches all the different user-agent versions of documents it encounters.
proxy.config.http.cache.vary_default_images STRING	NULL	Specifies the header on which Traffic Server varies for images.
proxy.config.http.cache.vary_default_other STRING	NULL	Specifies the header on which Traffic Server varies for anything other than text and images.
anonymous ftp password		
proxy.config.http.ftp.anonymous_passwd STRING		Specifies the anonymous password for FTP servers that require a password for access.
cached ftp document lifetime		
proxy.config.http.ftp.cache.document_lifetime INT	259200	Specifies the maximum amount of time that an FTP document can stay in the Traffic Server cache.
Customizable User Response Pages		
proxy.config.body_factory.enable_customizations INT	0	Specifies whether customizable response pages are enabled or disabled and which response pages are used. 0 = disable customizable user response pages 1 = enable customizable user response pages in the default directory only 2 = enable language-targeted user response pages
proxy.config.body_factory.enable_logging INT	1	Enables (1) or disables (0) logging for customizable response pages. When enabled, Traffic Server records a message in the error log each time a customized response page is used or modified.
proxy.config.body_factory.response_suppression_mode INT	0	Specifies when Traffic Server suppresses generated response pages. 0 = never suppress generated response pages 1 = always suppress generated response pages 2 = suppress response pages only for intercepted traffic
NNTP Engine		
proxy.config.nntp.enabled INT	0	Enables (1) or disables (0) Traffic Server from serving NNTP requests.

Configuration Variable Data Type	Default Value	Description
proxy.config.nntp.cache_enabled INT	1	Enables (1) or disables (0) Traffic Server from caching NNTP requests.
proxy.config.nntp.posting_enabled INT	1	Enables (1) or disables (0) posting. When enabled, users can post NNTP articles to parent NNTP servers.
proxy.config.nntp.access_control_enabled INT	0	Enables (1) or disables (0) access control. When enabled, you can control user access to news articles cached by Traffic Server according to the access privileges set in the nntp_access.config file (refer to *nntp_access.config, on page 251*).
proxy.config.nntp.v2_authentication INT	0	Enables (1) or disables (0) support for NNTP version 2. Enable this option only if you are certain that all your client authentication supports NNTP version 2.
proxy.config.nntp.cluster_enabled INT	1	Enables (1) or disables (0) cluster-wide NNTP caching.
proxy.config.nntp.feed_enabled INT	1	Enables (1) or disables (0) the allow feeds option that allows Traffic Server to accept feeds of news articles from feed or push groups.
proxy.config.nntp.logging_enabled INT	1	Enables (1) or disables (0) logging of NNTP transactions in the event logs.
roxy.config.nntp.background_posting_enabled INT	0	Enables (1) or disables (0) background posting. When enabled, Traffic Server posts NNTP articles to parent NNTP servers in the background.
proxy.config.nntp.insert_posting_trace_header INT	1	When enabled (1), Traffic Server inserts posting trace headers.
proxy.config.nntp.posting_ok_message STRING	`Inktomi NNTP server ready. posting ok`	Specifies the message that displays to news readers when they connect to the Traffic Server if posting is enabled.
proxy.config.nntp.posting_not_ok_message STRING	`Inktomi NNTP server ready. no posting`	Specifies the message that displays to news readers when they connect to the Traffic Server if posting is *not* enabled.
proxy.config.nntp.server_port INT	119	Specifies the port that Traffic Server uses to serve NNTP requests.
proxy.config.nntp.authorization_hostname STRING	NULL	Specifies the hostname of the authentication server. The value `NULL` defaults to localhost.
proxy.config.nntp.authorization_port INT	0	Specifies the port used for Traffic Server and authentication server communication.
proxy.config.nntp.obey_control_cancel INT	0	Enables (1) or disables (0) the obey cancel control messages option. When enabled, Traffic Server deletes the article from the cache when it receives a cancel control message.
proxy.config.nntp.obey_control_newgroup INT	0	Enables (1) or disables (0) the obey newgroups control messages option.

Configuration Variable Data Type	Default Value	Description
proxy.config.nntp.obey_control_rmgroup INT	0	Enables (1) or disables (0) the obey rmgroups control messages option.
proxy.config.nntp.inactivity_timeout INT	600	Specifies the number of seconds that idle NNTP connections can remain open. Inktomi recommends that you specify a value of at least 3 minutes.
proxy.config.nntp.check_newgroups_every INT	86400	Specifies how often (in seconds) Traffic Server polls parent NNTP servers for new groups. You must list the NNTP servers you want to poll in the `nntp_servers.config` file (refer to *nntp_servers.config, on page 252*).
proxy.config.nntp.check_newnews_every INT	900	Specifies how often (in seconds) Traffic Server checks new news on the NNTP servers.
proxy.config.nntp.check_cancels_every INT	3600	Specifies how often (in seconds) Traffic Server polls parent NNTP servers for canceled articles.
proxy.config.nntp.maintain_every INT	120	Specifies how often Traffic Server checks NNTP activities.
proxy.config.nntp.check_pull_every INT	600	Specifies how often Traffic Server caches news articles form pull groups.
proxy.config.nntp.group_check_parent_every INT	300	Specifies how often Traffic Server polls parent NNTP servers for new articles.
proxy.config.nntp.group_check_parent_adapt INT	128	When enabled (1), Traffic Server adapts to the article arrival rate by allowing checks to be done up to *n* times the base rate when no articles are seen.
proxy.config.nntp.group_check_cluster_every INT	60	Specifies how often Traffic Server polls other nodes in the cluster for new articles.
proxy.config.nntp.group_sync_every INT	600	Specifies how often Traffic Server synchronizes articles in memory to the cache.
proxy.config.nntp.group_expire_every INT	28800	Specifies how often Traffic Server checks for articles that have been collected by garbage collectors and clears them from the overview.
proxy.config.nntp.overview_sync_every INT	120	Specifies how often Traffic Server synchronizes overviews in memory to the cache.
proxy.config.nntp.overview_gc_every INT	1200	Specifies how often Traffic Server garbage collectors collect overviews in memory.
proxy.config.nntp.load_overview_min INT	25	Specifies the minimum number of overviews Traffic Server fetches at a time.
proxy.config.nntp.server_retry_timeout INT	60	Specifies how long Traffic Server must wait before retrying an origin server that was previously unavailable.

Configuration Variable Data Type	Default Value	Description
proxy.config.nntp.client_speed_throttle INT	0	Specifies the number of bytes per second that clients are allowed to download. A value of 0 means that downloading is not limited.
proxy.config.nntp.max_articles_per_group INT	100000	Specifies the maximum number of articles allowed per group.
proxy.config.nntp.forward_every INT	5	Specifies how often Traffic Server checks to see if articles need to be forwarded.
proxy.config.nntp.forward_backlog INT	1000	Specifies the number of forwarded articles to buffer.
proxy.config.nntp.add_to_path INT	0	When enabled (1), Traffic Server is added to the path header in articles.
proxy.config.nntp.forward_feed_only INT	0	When enabled (1), Traffic Server does not store feed articles locally, but forwards them.
proxy.config.nntp.auth_on_connect INT	0	When enabled (1), configures Traffic Server to signal the NNTP authentication plugin each time a client connects.
proxy.config.nntp.auth_on_disconnect INT	0	When enabled (1), configures Traffic Server to signal the NNTP authentication plugin each time a client disconnects.
proxy.config.nntp.highwind_auth_compat INT	0	Enables (1) or disables (0) highwind compatible authentication.
proxy.config.nntp.send_xref_in_overviews INT	1	Configures Traffic Server to send xrefs in overviews.
proxy.config.nntp.auth_server.binary STRING	nntp_auth	Specifies the file name of the NNTP authentication server plugin.
proxy.config.nntp.run_local_authentication_server INT	0	Runs the authentication server in local mode, enabling the traffic_cop process to watch and restart it automatically in case of failure.
proxy.config.nntp.accept_local_authentication_requests_only INT	1	When enabled (1), configures Traffic Server to only accept authentication requests from the same machine.
proxy.config.nntp.custom_authentication_via_stdio INT	0	When enabled (1), configures Traffic Server to pass authentication information via stdin instead of the environment variables.
proxy.config.nntp.first_authorization_ip STRING	0.0.0.0	Specifies the first IP address in the range of hosts permitted to connect to the authentication server.
proxy.config.nntp.last_authorization_ip STRING	255.255.255.255	Specifies the last IP address in the range of hosts permitted to connect to the authentication server.

Configuration Variable Data Type	Default Value	Description
	FTP Engine	
proxy.config.ftp.data_connection_mode INT	1	Specifies the FTP connection mode: 1 = PASV then PORT 2 = PORT only 3 = PASV only
proxy.config.ftp.control_connection_timeout INT	300	Specifies how long Traffic Server waits for a response from the FTP server.
proxy.config.ftp.ftp_enabled INT	0	Enables (1) or disables (0) processing of FTP requests from FTP clients.
proxy.config.ftp.cache_enabled INT	1	Enables (1) or disables (0) FTP documents to be put in the cache. If this option is disabled, Traffic Server always serves FTP documents from the FTP server.
proxy.config.ftp.logging_enabled INT	1	Enables (1) or disables (0) logging of FTP transactions.
proxy.config.ftp.proxy_server_port INT	21	Specifies the port used for FTP connections.
proxy.config.ftp.min_lisn_port INT	1024	Specifies the lowest port in the range of listening ports used by Traffic Server for data connections when the FTP client sends a PASV or Traffic Server sends a PORT to the FTP server.
proxy.config.ftp.max_lisn_port INT	65535	Specifies the highest port in the range of listening ports used by Traffic Server for data connections when the FTP client sends a PASV or Traffic Server sends a PORT to the FTP server.
proxy.config.ftp.server_data_default_pasv INT	1	Specifies the default method used to set up server side data connections. 1 = Traffic Server sends a PASV to the FTP server and lets the FTP server open a listening port. 0 = Traffic Server tries PORT first (sets up a listening port on the Traffic Server side of the connection).
proxy.config.ftp.try_pasv_times INT	1024	Specifies the number of times Traffic Server can try to open a listening port when the FTP client sends a PASV.
proxy.config.ftp.try_port_times INT	1024	Specifies the maximum number of times Traffic Server can try to open a listening port when sending a PORT to the FTP server.
proxy.config.ftp.try_server_ctrl_connect_times INT	6	Specifies the maximum number of times Traffic Server can try to connect to the FTP server's control listening port.
proxy.config.ftp.try_server_data_connect_times INT	3	Specifies the maximum number of times Traffic Server can try to connect to the FTP server's data listening port when it sends a PASV to the FTP server and gets the IP/listening port information.

Configuration Variable Data Type	Default Value	Description
proxy.config.ftp.try_client_data_connect_times INT	3	Specifies the maximum number of times Traffic Server can try to connect to the FTP client's data listening port when the FTP client sends a PORT with the IP/listening port information.
proxy.config.ftp.client_ctrl_no_activity_timeout INT	900	Specifies the no activity timeout for the FTP client control connection.
proxy.config.ftp.client_ctrl_active_timeout INT	14400	Specifies the active timeout for the FTP client control connection.
proxy.config.ftp.server_ctrl_no_activity_timeout INT	900	Specifies the inactivity timeout for the FTP server control connection.
proxy.config.ftp.server_ctrl_active_timeout INT	14400	Specifies the active timeout for the FTP server control connection.
proxy.config.ftp.pasv_accept_timeout INT	120	Specifies the timeout value for a listening data port in traffic server (for PASV, for the client data connection)
proxy.config.ftp.port_accept_timeout INT	120	Specifies the timeout value for a listening data port in Traffic Server (for PORT, for the server data connection)
proxy.config.ftp.share_ftp_server_ctrl_enabled INT	1	Enables (1) or disables (0) sharing the server control connections among multiple anonymous FTP clients.
proxy.config.ftp.server_ctrl_keep_alive_no_activity_timeout INT	90	Specifies the timeout value when the FTP server control connection is not used by any FTP clients.
proxy.config.ftp.reverse_ftp_enabled INT	0	Enables (1) or disables (0) the FTP reverse proxy option. If enabled, you must configure the ftp_remap.config file. *Setting FTP Mapping Rules, on page 74*.
proxy.config.ftp.login_info_fresh_in_cache_time INT	2592000	Specifies how long the 220/230 responses (login messages) can stay fresh in the cache.
proxy.config.ftp.directory_listing_fresh_in_cache_time INT	604800	Specifies how long directory listings can stay fresh in the cache.
proxy.config.ftp.file_fresh_in_cache_time INT	259200	Specifies how long FTP files can stay fresh in the cache.
proxy.config.ftp.simple_directory_listing_cache_enabled INT	1	Enables (1) or disables (0) caching of directory listings without arguments (for example, `dir/ls`).
proxy.config.ftp.full_directory_listing_cache_enabled INT	1	Enables (1) or disables (0) caching of directory listings with arguments (for example, `ls -al`, `ls *.txt`).
	SOCKS Processor	
proxy.config.socks.socks_needed INT	0	Enables (1) or disables (0) the SOCKS option. Refer to *Configuring SOCKS firewall integration, on page 142*.
proxy.config.socks.socks_version FLOAT	4.0	Specifies the SOCKS version.

Configuration Variable Data Type	Default Value	Description
proxy.config.socks.socks_server_ip_str STRING	0.0.0.0	Specifies the IP address of the SOCKS server.
proxy.config.socks.socks_server_port INT	1080	Specifies the port used by Traffic Server to communicate with the SOCKS server.
proxy.config.socks.socks_timeout INT	100	Specifies the number of seconds the Traffic Server must wait for the SOCKS server to respond before dropping the connection.
Net Subsystem		
proxy.config.net.connections_throttle INT	8000	Specifies the maximum number of connections that Traffic Server can handle. If Traffic Server receives additional client requests, they are queued until existing requests are served.
Cluster Subsystem		
proxy.config.cluster.cluster_port INT	8086	Specifies the port used for cluster communication.
proxy.config.cluster.ethernet_interface STRING	hme0	Specifies the network interface used for cluster traffic. All nodes in a cluster must use the same network interface. Solaris only.
Cache		
proxy.config.cache.permit.pinning INT	0	Enables (1) or disables (0) the cache pinning option.
proxy.config.cache.ram_cache.size INT	-1	Specifies the size of the RAM cache in MB. -1 = the RAM cache is automatically sized at approximately 1 MB per GB of disk.
proxy.config.cache.limits.http.max_alts INT	3	Specifies the maximum number of HTTP alternates that Traffic Server can cache.
proxy.config.cache.max_doc_size INT	0	Specifies the maximum size of documents in the cache. 0 = there is no size limit.
DNS		
proxy.config.dns.search_default_domains INT	1	Enables (1) or disables (0) local domain expansion so that Traffic Server can attempt to resolve unqualified hostnames by expanding to the local domain. For example, if a client makes a request to an unqualified host named `host_x`, and if the Traffic Server's local domain is `y.com`, the Traffic Server will expand the hostname to `host_x.y.com`.
proxy.config.dns.splitDNS.enabled INT	0	Enables (1) or disables (0) DNS server selection. When enabled, Traffic Server refers to the splitdns.config file for the selection specification. Refer to *Configuring DNS server selection (split DNS), on page 145*.

Configuration Variable Data Type	Default Value	Description
proxy.config.dns.splitdns.def_domain STRING	NULL	Specifies the default domain for split DNS requests. This value is appended automatically to the hostname if it does not include a domain before split DNS determines which DNS server to use.
proxy.config.dns.url_expansions STRING	NULL	Specifies a list of hostname extensions that are automatically added to the hostname after a failed lookup. For example, if you want Traffic Server to add the hostname extension .org, specify `org` as the value for this variable (Traffic Server automatically adds the dot (.). Note: If the variable proxy.config.http.enable_url_expandomatic is set to 1 (the default value), you do not have to add www. and .com to this list; Traffic Server tries www. and .com automatically after trying the values you specify.
	HostDB	
proxy.config.hostdb.size INT	200000	Specifies the maximum number of entries allowed in the host database.
proxy.config.hostdb.ttl_mode INT	0	Specifies the host database time to live mode. You can specify one of the following: 0 = obey 1 = ignore 2 = min(X,ttl) 3 = max(X,ttl)
proxy.config.hostdb.timeout INT	1440	Specifies the foreground timeout, in seconds.
proxy.config.hostdb.strict_round_robin INT	0	When disabled (0), Traffic Server always uses the same origin server for the same client as long as the origin server is available.
	Logging Config	
proxy.config.log2.logging_enabled INT	3	Enables and disables event logging: 0 = logging disabled 1 = log errors only 2 = log transactions only 3 = full logging (errors + transactions) Refer to *Chapter 12, Working with Log Files*.
proxy.config.log2.max_secs_per_buffer INT	5	Specifies the maximum amount of time before data in the buffer is flushed to disk.
proxy.config.log2.max_space_mb_for_logs INT	100	Specifies the amount of space allocated to the logging directory in MB.
proxy.config.log2.max_space_mb_for_orphan_logs INT	25	Specifies the amount of space allocated to the logging directory in MB if this node is acting as a collation client.

Configuration Variable Data Type	Default Value	Description
proxy.config.log2.max_space_mb_headroom INT	10	Specifies the tolerance for the log space limit in bytes. If the variable proxy.config.log2.auto_delete_rolled_file is set to 1 (enabled), autodeletion of log files is triggered when the amount of free space available in the logging directory is less than the value specified here.
proxy.config.log2.hostname STRING	localhost	Specifies the hostname of the machine running Traffic Server.
proxy.config.log2.logfile_dir STRING	`install_dir/logs`	Specifies the full path to the logging directory.
proxy.config.log2.logfile_perm STRING	rw-r--r--	Specifies the log file permissions. The standard Unix file permissions are used (owner, group, other). Valid values are: `-` no permission `r` read permission `w` write permission `x` execute permission Note: Permissions are subject to the umask settings for the traffic server process. This means that a umask setting of 002 will not allow write permission for others, even if specified in the configuration file. Permissions for existing log files are not changed when the configuration is changed. UNIX only.
proxy.config.log2.custom_logs_enabled INT	0	Enables (1) or disables (0) custom logging.
proxy.config.log2.xml_logs_config INT	0	Enables (1) or disables (0) extended custom logging using an XLM-based configuration file. A value of 0 instructs Traffic Server to use the traditional custom log formats.
proxy.config.log2.squid_log_enabled INT	1	Enables (1) or disables (0) the squid log file format.
proxy.config.log2.squid_log_is_ascii INT	1	Specifies the squid log file type. 1 = ASCII 0 = binary
proxy.config.log2.squid_log_name STRING	squid	Specifies the squid log file name.
proxy.config.log2.squid_log_header STRING	NULL	Specifies the squid log file header text.
proxy.config.log2.common_log_enabled INT	0	Enables (1) or disables (0) the Netscape common log file format.

Configuration Variable Data Type	Default Value	Description
proxy.config.log2.common_log_is_ascii INT	1	Specifies the Netscape common log file type. 1 = ASCII 0 = binary
proxy.config.log2.common_log_name STRING	common	Specifies the Netscape common log file name.
proxy.config.log2.common_log_header STRING	NULL	Specifies the Netscape common log file header text.
proxy.config.log2.extended_log_enabled INT	0	Enables (1) or disables (0) the Netscape extended log file format.
proxy.confg.log2.extended_log_is_ascii INT	1	Specifies the Netscape extended log file type. 1 = ASCII 0 = binary
proxy.config.log2.extended_log_name STRING	extended	Specifies the Netscape extended log file name.
proxy.config.log2.extended_log_header STRING	NULL	Specifies the Netscape extended log file header text.
proxy.config.log2.extended2_log_enabled INT	0	Enables (1) or disables (0) the Netscape extended-2 log file format.
proxy.config.log2.extended2_log_is_ascii INT	1	Specifies the Netscape extended-2 log file type. 1 = ASCII 0 = binary
proxy.config.log2.extended2_log_name STRING	extended2	Specifies the Netscape extended-2 log file name.
proxy.config.log2.extended2_log_header STRING	NULL	Specifies the Netscape extended-2 log file header text.
proxy.config.log2.separate_icp_logs INT	0	When enabled (1), configures Traffic Server to store ICP transactions in a separate log file.
proxy.config.log2.separate_nntp_logs INT	0	When enabled (1), configures Traffic Server to store NNTP transactions in a separate log file.
proxy.config.log2.separate_mixt_logs INT	0	When enabled (1), configures Traffic Server to store streaming media transactions in a separate log file. Refer to the *Media-IXT User's Guide*.
proxy.config.log2.separate_host_logs INT	0	When enabled (1), configures Traffic Server to create a separate log file for HTTP/FTP transactions for each origin server listed in the `log_hosts.config` file (refer to *HTTP host log splitting, on page 170*).

Configuration Variable Data Type	Default Value	Description
proxy.local.log2.collation_mode INT	0	Specifies the log collation mode: 0 = Collation disabled. 1 = This host is a log collation server. 2 = This host is a collation client, and sends entries using standard formats to the collation server. 3 = This host is a collation client, and sends traditional custom formats to the collation server. 4 = This host is a collation client, and sends both standard and traditional custom formats to the collation server. For information on sending XML-based custom formats to the collation server, refer to *logs_xml.config, on page 245*.
proxy.confg.log2.collation_host STRING	NULL	Specifies the hostname of the log collation server.
proxy.config.log2.collation_port INT	8085	Specifies the port used for communication between the collation server and client.
proxy.config.log2.collation_secret STRING	foobar	Specifies the password used to validate logging data and prevent the exchange of unauthorized information when a collation server is being used.
proxy.config.log2.collation_host_tagged INT	0	When enabled (1), configures Traffic Server to include the hostname of the collation client that generated the log entry in each entry.
proxy.config.log2.collation_retry_sec INT	5	Specifies the number of seconds between collation server connection retries.
proxy.config.log2.rolling_enabled INT	1	Enables (1) or disables (0) log file rolling. Refer to *Rolling event log files, on page 167*.
proxy.config.log2.rolling_interval_sec INT	86400	Specifies the log file rolling interval, in seconds. The minimum value is 300 (5 minutes).
proxy.config.log2.rolling_offset_hr INT	0	Specifies the file rolling offset hour. The hour of the day that starts the log rolling period.
proxy.config.log2.auto_delete_rolled_files INT	1	Enables (1) or disables (0) automatic deletion of rolled files.
proxy.config.log2.sampling_frequency INT	1	Configures Traffic Server to log only a sample of transactions rather than every transaction. You can specify the following values: 1 = log every transaction 2 = log every second transaction 3 = log every third transaction and so on...

Configuration Variable Data Type	Default Value	Description
	QuickTime Config	
proxy.config.qt.tcp_to_server INT	1	Refer to the Media-IXT User's Guide.
proxy.config.qt.proxy_port INT	1091	Refer to the Media-IXT User's Guide.
	RNI Config	
proxy.config.rni.watcher_enabled INT	0	Refer to the Media-IXT User's Guide.
proxy.config.rni.proxy_rtsp_port INT	6060	Refer to the Media-IXT User's Guide.
proxy.config.rni.proxy_port INT	1091	Refer to the Media-IXT User's Guide.
proxy.config.rni.proxy_pid_path STRING	NULL	Refer to the Media-IXT User's Guide.
proxy.config.rni.proxy_restart_cmd STRING	NULL	Refer to the Media-IXT User's Guide.
proxy.config.rni.proxy_restart_interval INT	10	Refer to the Media-IXT User's Guide.
proxy.config.rni.proxy_service_name STRING	RMProxy	Refer to the Media-IXT User's Guide.
	Reverse Proxy	
proxy.config.reverse_proxy.enabled INT	0	Enables (1) or disables (0) reverse proxy. (HTTP only). For FTP reverse proxy, refer to *proxy.config.ftp.reverse_ftp_enabled, on page 277*.
URL Remap Rules		
proxy.config.url_remap.default_to_server_pac INT	0	Enables (1) or disables (0) requests for / and /proxy.pac on the proxy port to be sent to the PAC port.
proxy.config.url_remap.default_to_server_pac_port INT	-1	Sets the PAC port:. -1 specifies that the PAC port will be set to the auto-configuration port. If you specify a specific port, PAC requests made to Traffic Server are redirected this port.
proxy.config.url_remap.remap_required INT	0	Set this variable to 1 if you want Traffic Server to serve requests only from origin servers listed in the mapping rules of the remap.config file. If a request does not match, the browser will receive an error.
proxy.config.url_remap.pristine_host_hdr INT	0	Set this variable to 1 if you want to retain the client host header in a request during remapping.

Configuration Variable Data Type	Default Value	Description
	SSL Termination	
proxy.config.ssl.enabled INT	0	Enables (1) or disables (0) the SSL termination option. Refer to *Using SSL Termination, on page 147*.
proxy.config.ssl.server_port INT	4443	Specifies the port used for SSL communication.
proxy.config.ssl.client.certification_level INT	0	Sets the client certification level: `0` = no client certificates are required. Traffic Server does not verify client certificates during the SSL handshake. Access to Traffic Server depends on Traffic Server configuration options (such as access control lists). `1` = client certificates are optional. If a client has a certificate, the certificate is validated. If the client does not have a certificate, the client is still allowed access to Traffic Server unless access is denied through other Traffic Server configuration options. `2` = client certificates are required. The client must be authenticated during the SSL handshake. Clients without a certificate are not allowed to access Traffic Server.
proxy.config.ssl.server.cert.filename STRING	server.pem	Specifies the file name of Traffic Server's SSL certificate (the server certificate).
proxy.config.ssl.server.cert.path STRING	/config	Specifies the location of Traffic Server's SSL certificate (the server certificate).
proxy.config.ssl.server.private_key.filename STRING	NULL	Specifies the file name of Traffic Server's private key. Change this variable only if the private key is not located in the Traffic Server's SSL certificate file.
proxy.config.ssl.server.private_key.path STRING	NULL	Specifies the location of the Traffic Server's private key. Change this variable only if the private key is not located in the SSL certificate file.
proxy.config.ssl.CA.cert.filename STRING	NULL	Specifies the file name of the certificate authority that client certificates will be verified against.
proxy.config.ssl.CA.cert.path STRING	NULL	Specifies the location of the certificate authority file that client certificates will be verified against.
proxy.config.ssl.client.verify.server INT	0	Configures Traffic Server to verify the origin server certificate with the Certificate Authority (CA).
proxy.config.ssl.client.cert.filename STRING	NULL	Specifies the file name of SSL client certificate installed on Traffic Server.

Configuration Variable Data Type	Default Value	Description
proxy.config.ssl.client.cert.path STRING	/config	Specifies the location of the SSL client certificate installed on Traffic Server.
proxy.config.ssl.client.private_key.filename STRING	NULL	Specifies the file name of Traffic Server's private key. Change this variable only if the private key is not located in the Traffic Server's SSL client certificate file.
proxy.config.ssl.client.private_key.path STRING	NULL	Specifies the location of the Traffic Server's private key. Change this variable only if the private key is not located in the SSL client certificate file.
proxy.config.ssl.client.CA.cert.filename STRING	NULL	Specifies the file name of the certificate authority against which the origin server will be verified.
proxy.config.ssl.client.CA.cert.path STRING	NULL	Specifies the location of the certificate authority file against which the origin server will be verified.
	ICP Configuration	
proxy.config.icp.enabled INT	0	Sets ICP mode for hierarchical caching: 0 = disables ICP. 1 = allows Traffic Server to receive ICP queries only. 2 = allows Traffic Server to send and receive ICP queries. Refer to *ICP cache hierarchies, on page 95*.
proxy.config.icp.icp_interface STRING	hme0 (for Solaris only)	Specifies the network interface used for ICP traffic.
proxy.config.icp.icp_port INT	3130	Specifies the UDP port that you want to use for ICP messages.
proxy.config.icp.multicast_enabled INT	0	Enables (1) or disables (0) ICP multicast.
proxy.config.icp.query_timeout INT	2	Specifies the timeout used for ICP queries.
	Scheduled Update Configuration	
proxy.config.update.enabled INT	0	Enables (1) or disables (0) the Scheduled Update option.
proxy.config.update.force INT	0	Enables (1) or disables (0) a force immediate update. When enabled, Traffic Server overrides the scheduling expiration time for all scheduled update entries and initiates updates until this option is disabled.
proxy.config.update.retry_count INT	10	Specifies the number of times Traffic Server can retry the scheduled update of a URL in the event of failure.

Configuration Variable Data Type	Default Value	Description
proxy.config.update.retry_interval INT	2	Specifies the delay in seconds between each scheduled update retry for a URL in the event of failure.
proxy.config.update.concurrent_updates INT	100	Specifies the maximum simultaneous update requests allowed at any point in time. This option prevents the scheduled update process from overburdening the host.
	SNMP Configuration	
proxy.config.snmp.master_agent_enabled INT	1	Enables (1) or disables (0) the SNMP agent.
	Plug-in Configuration	
proxy.config.plugin.plugin_dir STRING	config/plugins	Specifies the location of Traffic Server plugins.
	WCCP Configuration	
proxy.config.wccp.enabled INT	0	Enables (1) or disables (0) WCCP.
proxy.config.wccp.version INT	1	Specifies the version of WCCP being used. 1 = Version 1.0. 2 = Version 2.0.
WCCP 1.0 variables		
proxy.config.wccp.router_ip STRING	NULL	Specifies the IP address of the router sending traffic to Traffic Server.
proxy.config.wccp.ethernet_interface STRING	NULL	Specifies the interface used to talk to the router.
WCCP 2.0 variables		
proxy.config.wccp2.security_enabled INT	0	Enables (1) or disables (2) security so that the router and the Traffic Server can authenticate each other. (If you enable security in Traffic Server, you must also enable security on the router, refer to your Cisco router documentation.)
proxy.config.wccp2.password STRING	NULL	Specifies the password used for authentication. This must be the same password configured on the router. It must be at least seven characters long.
proxy.config.wccp2.multicast_enabled INT	0	Enables (1) or disables (2) multicast mode.
proxy.config.wccp2.multicast_address STRING	NULL	Specifies the IP multicast address.
proxy.config.wccp2.number_of_routers INT	0	If multicast is *not* enabled, the routers on your network are not automatically discovered. You must specify the number of routers that direct traffic to Traffic Server. WCCP 2.0 supports a maximum of 32 routers.

Configuration Variable Data Type	Default Value	Description
proxy.config.wccp2.router0_ip STRING ... proxy.config.wccp2.router10_ip STRING	NULL	If multicast is *not* enabled, the routers on your network are not automatically discovered. You must specify the IP address of each router that directs traffic to Traffic Server.
proxy.config.wccp2.svc_HTTP INT	1	Enables (1) or disables (0) transparent redirection of HTTP requests.
proxy.config.wccp2.svc_NNTP INT	1	Enables (1) or disables (0) transparent redirection of NNTP requests.
proxy.config.wccp2.svc_RTSP INT	1	Refer to the *Media-IXT User's Guide*.
proxy.config.wccp2.svc_PNA INT	1	Refer to the *Media-IXT User's Guide*.
proxy.config.wccp2.svc_WMT INT	1	Refer to the *Media-IXT User's Guide*.
proxy.config.wccp2.svc_FTP INT		Enables (1) or disables (0) transparent redirection of FTP requests from FTP clients.
proxy.local.wccp2.ethernet_interface STRING	hme0	Specifies the interface used to talk to the router.
	ARM (Security Configuration)	
proxy.config.arm.security_enabled INT	0	Enables (1) or disables (0) ARM security. Refer to *Controlling host access to the Traffic Server machine (ARM security), on page 134*.

remap.config

The remap.config file contains mapping rules that Traffic Server uses to:

- ✔ Map URL requests for a specific origin server to the appropriate location on Traffic Server when Traffic Server acts as a reverse proxy for that particular origin server
- ✔ Reverse-map server location headers so that when origin servers respond to a request with a location header that redirects the client to another location, the clients do not bypass the Traffic Server
- ✔ Redirect HTTP requests permanently or temporarily without Traffic Server having to contact any origin servers

Refer to *Chapter 5, Reverse Proxy and HTTP Redirects* for information about redirecting HTTP requests and using reverse proxy.

Important After you modify the remap.config file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's bin directory your working directory, then run the traffic_line -x command. In Windows, open a Command Prompt window, cd to the bin directory (located in the Traffic Server installation directory), then run the traffic_line -x command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line in the remap.config file must contain a mapping rule. Traffic Server recognizes three space-delimited fields: type, target, and replacement. The following table describes the format of each field.

Field	Description
type	Enter either one of the following: map—translates an incoming request URL to the appropriate origin server URL (HTTP reverse proxy). reverse_map—translates the URL in origin server redirect responses to point to the Traffic Server (HTTP reverse proxy). redirect—redirects HTTP requests permanently without having to contact the origin server. Permanent redirects notify the browser of the URL change (by returning an HTTP status code 307) so that the browser can update bookmarks. redirect_temporary—redirects HTTP requests temporarily without having to contact the origin server. Temporary redirects notify the browser of the URL change for the current request only (by returning an HTTP status code 301).
target	Enter the origin or *from* URL. You can enter up to four components: *scheme*://*host*:*port*/*path_prefix*
replacement	Enter the destination or *to* URL. You can enter up to four components: *scheme*://*host*:*port*/*path_prefix*

Examples

The following section shows example mapping rules in the `remap.config` file.

Reverse proxy mapping rules

The following example shows a map rule that does not specify a path prefix in the target or replacement:

```
map http://www.x.com/ http://server.hoster.com
```

This rule results in the following translations:

Client Request	Translated Request
http://www.x.com/Widgets/index.html	http://server.hoster.com/Widgets/index.html
http://www.x.com/cgi/form/submit.sh?arg=true	http://server.hoster.com/cgi/form/submit.sh?arg=true

The following example shows a map rule with path prefixes specified in the target:

```
map http://intranet.y.com/marketing http://marketing.y.com/
```

```
map http://intranet.y.com/sales http://sales.y.com
```

```
map http://intranet.y.com/engineering http://engineering.y.com/
```

```
map http://intranet.y.com/ http://info.y.com/
```

These rules result in the following translations:

Client Request	Translated Request
http://www.y.com/marketing/projects/manhattan/specs.html	http://marketing.y.com/projects/manhattan/specs.html
http://www.y.com/stuff/marketing/projects/boston/specs.html	http://info.y.com/marketing/projects/boston/specs.html
http://www.y.com/engineering/marketing/requirements.html	http://engineering.y.com/marketing/requirements.html

The following example shows that the order of the rules matters:

```
map http://www.g.com/ http://external.g.com/
```

```
map http://www.g.com/stuff http://stuff.g.com
```

The above rules result in the following translation:

Client Request	Translated Request
http://www.g.com/stuff/a.gif	http://external.g.com/stuff/a.gifl

In the above examples, the second rule is never applied because all URLs that match the second rule also match the first rule. The first rule takes precedence because it appears earlier in the `remap.config` file.

The following example shows a mapping with a path prefix specified in the target and replacement:

```
map http://www.h.com/a/b http://server.h.com/customers/x/y
```

This rule results in the following translation:

Client Request	Translated Request
http://www.h.com/a/b/c/d/doc.html	http://server.h.com/customers/x/y/c/d/doc.html
http://www.h.com/a/index.html	Translation fails

The following example shows reverse-map rules:

```
map http://www.x.com/ http://server.hoster.com/x/
```

```
reverse_map http://server.hoster.com/x/ http://www.x.com/
```

These rules result in the following translations:

Client Request	Translated Request
http://www.x.com/Widgets	http://server.hoster.com/x/Widgets

Client Request	Origin server Header	Translated Header
http://www.x.com/Widgets	http://server.hoster.com/x/Widgets/	http://www.x.com/Widgets/

Note When acting as a reverse proxy for multiple servers, the Traffic Server is unable to route to URLs from older browsers that do not send the `Host:` header. As a solution, set the **Redirect requests without host header to URL** option in the **Mapping/Redirection** section of the **Routing** page of the Traffic Manager UI to a page that explains the situation to the user and advises a browser upgrade or provides a link directly to the origin server, bypassing the Traffic Server.

Redirect mapping rules

The following rule permanently redirects all HTTP requests for `www.inktomi` to `www.inktomi2.com`.

```
redirect http://www.inktomi.com http://www.inktomi2.com
```

The following rule temporarily redirects all HTTP requests for `www.inktomi` to `www.inktomi3.com`.

```
redirect_temporary http://www.inktomi.com http://www.inktomi2.com
```

snmpd.cnf

The snmpd.cnf file contains parameters that control user access to MIB information and trap destinations. It is beyond the scope of this manual to describe all of the SNMP parameters and formats; only the major parameters affecting access control and trap destination are discussed in this section.

Important After you modify the snmpd.cnf file, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's bin directory your working directory, then run the traffic_line -x command. In Windows, open a Command Prompt window, cd to the bin directory (located in the Traffic Server installation directory), then run the traffic_line -x command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

The snmpd.cnf file contains a list of configuration parameters. As with all other configuration files, lines beginning with the # symbol are comments. Each configuration parameter is listed along with formatting variables, as in the following example:

```
#Entry type: snmpNotifyEntry
#Format:  snmpNotifyName  (text)
#         snmpNotifyTag  (text)  (keyed on snmpTargetAddr table)
#         snmpNotifyType  (trap(1), inform(2))
#         snmpNotifyStorageType  (nonVolatile, permanent, readOnly)
#snmpNotifyEntry  31 Console trap nonVolatile
#snmpNotifyEntry  32 TrapSink trap nonVolatile
```

Configuring trap destinations

You must modify the snmpd.cnf file to send traps to each of your monitoring stations.

You must configure the snmpnotifyEntry and snmpTargetAddrEntry entries for trap destinations. snmpnotifyEntry sends traps to a particular host or group of hosts. snmpTargetAddrEntry defines the IP addresses for a host or group of hosts.

For example, to send traps to a host named host_a, you need an snmpnotifyEntry line similar to the following:

```
snmpnotifyEntry 31 host_a trap nonVolatile
```

This line defines a trap destination named host_a which can represent a single IP address or a group of IP addresses. In place of *host_a*, enter the name of the host or group of hosts to receive traps on your system. In place of *31*, enter a unique integer.

Then, for each IP address that you want to define for host_a, you must enter a snmpTargetAddrEntry line similar to the following. All trap messages destined for host_a are sent to the IP addresses defined in the snmpTargetAddrEntry lines of the snmpd.cnf file.

```
snmpTargetAddrEntry 34 snmpUDPDomain A.B.C.D:0 100 3 host_a v1ExampleParams nonVolatile 255.255.255.255:0
```

In place of *34*, enter a unique integer. In place of *A.B.C.D*, enter the IP address that you want to define for host_a.

Access control

By default, read-only access is granted to any host that makes SNMP requests using the community string public. To restrict access, you need to remove access-related default entries in the snmpd.cnf file and add entries specifying the hosts you want to allow. You must:

- ✔ Define the hosts or host groups for your system (use the `snmpTargetAddrEntry` lines to define the IP addresses associated to each host or host group)
- ✔ Define access communities (a community can consist of a host or group of hosts); you need to define hosts before you can define communities
- ✔ Give access to the communities that you want to have access; you need to define communities in order to give them access

Examples

To restrict access, remove the following default `snmpd.cnf` entries, which allow access to any host:

```
vacmAccessEntry snmpv1 public Anyone nonVolatile
```

```
vacmAccessEntry snmpv2c public Anyone nonVolatile
```

```
snmpCommunityEntry t0000000 public public localSnmpID - nonVolatile
```

To allow access to selected hosts, replace the deleted entries with the following. You can allow access to as many hosts as you want. You can configure one host at a time or one subnet at a time.

For example, suppose you want to allow the single host named `OneHost` to have access to MIB information. You would need the following lines in the `snmpd.cnf` file:

```
snmpTargetAddrEntry 33 snmpUDPDomain A.B.C.D:0 100 3 host_a v1ExampleParams nonVolatile 255.255.255.255:0
```

```
snmpCommunityEntry localSnmpID public OneHost localSnmpID default host_a nonVolatile
```

```
vacmAccessEntry OneHost - snmpv1 noAuthNoPriv exact All - All nonVolatile
```

```
vacmSecurityToGroupEntry snmpv1 public OneHost nonVolatile
```

The `snmpTargetAddrEntry` line defines the host, `host_a`, which has the IP address `A.B.C.D`. The `communityEntry` line defines the community `OneHost`, which contains the host `host_a`. The `vacmAccessEntry` and `vacmSecurityToGroupEntry` lines allow access to the community `OneHost`.

To allow MIB access to one subnet named `OneNet`, enter the following lines in the configuration file:

Note Use the netmask `255.255.255.0` for the subnet `A.B.C.xxx` in the `snmpTargetAddrEntry` definition.

```
snmpTargetAddrEntry 34 snmpUDPDomain A.B.C.0:0 100 3 net_a v1ExampleParams nonVolatile 255.255.255.0:0
```

```
communityEntry localSnmpID public OneNet localSnmpID default net_a nonVolatile
```

```
vacmAccessEntry OneNet - snmpv1 no AuthNoPriv exact All - All nonVolatile
```

```
vacmSecurityToGroupEntry snmpv1 public OneNet nonVolatile
```

The `snmpTargetAddrEntry` line defines the subnet, `net_a`, which has the IP address `A.B.C.xxx`. The `communityEntry` line defines the community `OneNet`, which contains the subnet `net_a`. The `vacmAccessEntry` and `vacmSecurityToGroupEntry` lines allow access to the community `OneNet`.

socks.config

The socks.config file specifies which origin servers you want Traffic Server to access directly without going through the SOCKS server.

Important After you modify socks.config, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's bin directory your working directory, then run the traffic_line -x command. In Windows, open a Command Prompt window, cd to the bin directory (located in the Traffic Server installation directory), then run the traffic_line -x command.

If you are running a cluster, you need only run the command for one node; the changes will propagate.

Format

Each line in the socks.config file can consist of a maximum of 400 characters. Any line not containing no_socks as the first string will be ignored. Any line containing no_socks as the first string must follow the format below:

```
no_socks IPaddresses_or_IPaddress_range
```

where *IPaddresses_or_IPaddress_range* is a comma separated list of the IP addresses or IP address ranges associated with the origin servers you want Traffic Server to access directly.

Examples

The following example configures Traffic Server to access the origin server associated with the IP address 11.11.11.1 directly without going through the SOCKS server.

```
no_socks 11.11.11.1
```

The following example configures Traffic Server to access the origin servers associated with the range of IP addresses 123.14.15.1 - 123.14.17.4 and the IP address 113.14.18.2 directly without going through the SOCKS server.

```
no_socks 123.14.15.1 - 123.14.17.4, 113.14.18.2
```

splitdns.config

The `splitdns.config` file enables you to specify the DNS server that Traffic server should use for resolving hosts under specific conditions.

To specify a DNS server, you must supply the following information in each active line within the file:

- ✔ A primary destination specifier in the form of a destination domain, a destination host, or a URL regular expression
- ✔ A set of server directives, listing one or more DNS servers with corresponding port numbers

You may also include the following optional information with each DNS server specification:

- ✔ A default domain for resolving hosts
- ✔ A search list specifying the domain search order when multiple domains are specified

For more information, refer to *Configuring DNS server selection (split DNS), on page 145*.

Format

Each line in the `splitdns.config` file uses one of the following formats:

```
dest_domain=dest_domain | dest_host | url_regex named="dns_server"
def_domain="def_domain" search_list="search_list"
```

The following table describes each field:

Field	Allowed inputs
dest_domain	A valid domain name. This specifies that the DNS server selection be based on the destination domain. You can prefix the domain with the "!" symbol to indicate the NOT logical operator.
dest_host	A valid hostname. This specifies that the DNS server selection be based on the destination host. You can prefix the host with the "!" symbol to indicate the NOT logical operator.
url_regex	A valid URL regular expression. This specifies that the DNS server selection be based on a regular expression.
dns_server	This is a required directive. It identifies the DNS server for Traffic Server to use with the given destination specifier. You can specify a port using the ":" symbol; if you don't supply a port, 53 is used. You can specify multiple DNS servers separated by spaces or the ";" symbol. Note: You must specify the domains using IP addresses in dot notation.
def_domain	A valid domain name. This optional directive specifies the default domain name to use for resolving hosts. Only one entry is allowed. If you do not provide the default domain, the system determines its value from /etc/resolv.conf on UNIX, or from the Registry on Windows platforms.
search_list	A list of domains separated by spaces or ";". This specifies the domain search order. If you do not provide the search list, the system determines the value from /etc/resolv.conf on UNIX, or from the Registry on Windows platforms.

Examples

Consider the following DNS server selection specifications:

```
dest_domain=internal.inktomi.com named=“255.255.255.255:212
255.255.255.254” def_domain=“inktomi.com” search_list=“inktomi.com
inktomi1.com”
```

```
dest_domain=!internal.inktomi.com named=“255.255.255.253”
```

Now consider the following two requests:

- ✔ http://minstar.internal.inktomi.com

  This request will match the first line and select DNS server `255.255.255.255` on port `212`. All resolver requests will use `inktomi.com` as the default domain, and `inktomi.com` and `inktomi1.com` as the set of domains to search first.

- ✔ http://www.microsoft.com

  This request will match the second line, namely not `internal.inktomi.com`. Therefore Traffic Server selects DNS server `255.255.255.253`. Since, no `def_domain` or `search_list` was supplied, Traffic Server retrieves this information from `/etc/resolv.conf` on UNIX, or from the Registry on Windows platforms.

storage.config

The storage.config file lists all the files, directories, or hard disk partitions that make up the Traffic Server cache.

Important After you modify storage.config, the Traffic Manager has to reread the configuration files. In UNIX, make Traffic Server's bin directory your working directory, then run the traffic_line -x command. In Windows, open a Command Prompt window, cd to the bin directory (located in the Traffic Server installation directory), then run the traffic_line -x command.

Format

The format of the storage.config file is:

```
pathname size
```

Where *pathname* is the name of a partition, directory, or file, and *size* is the size of the named partition, directory, or file, in bytes. You must specify a size for directories or files. For raw partitions, size specification is optional.

You can use any partition of any size. For best performance, Inktomi recommends the following:

✔ Use raw disk partitions

✔ For each disk, make all partitions the same size

✔ For each node, use the same number of partitions on all disks

Specify pathnames according to your operating system requirements. See the following examples.

Examples

The following basic example shows 64 MB of cache storage in the /big_dir directory:

```
/big_dir 67108864
```

You can use the . symbol for the current directory. Here is an example of 64 MB of cache storage in the current directory:

```
. 67108864
```

Solaris example

The following example is for the Solaris operating system:

```
/devices/sbus@1f,0/SUNW,fas@e,880000/sd@2,0:a,raw
/devices/sbus@1f,0/SUNW,fas@e,880000/sd@2,0:b,raw
```

Note The size is not required, because the partitions are raw.

Digital UNIX example

Here is an example for a Digital UNIX configuration:

```
/dev/rrz14c 67108864
```

Windows example

Here is an example for a Windows configuration:

```
D:\TrafficServer\2.3\cache 67108864
```

update.config

The `update.config` file controls how Traffic Server performs a scheduled update of specific local cache content. The file contains a list of URLs specifying objects that you want to schedule for update. Be aware that the system validates these URLs for syntactic correctness, but not for existence.

Conceptually, a scheduled update performs a local HTTP `GET` on the objects at the specific time or interval. You can control the following parameters for each specified object:

✔ The URL

✔ URL-specific Request Headers, which overrides the default

✔ The update time and interval

✔ The recursion depth

Note Always use the **Content Management** page to modify settings in the `update.config` file instead of modifying the file directly using a text editor. This is recommended because certain fields contain special characters, such as <CR><LF>. In addition, the **Content Management** page automatically notifies Traffic Server when changes are made. If you do modify the `update.config` file directly, you must manually stop and restart the Traffic Server process.

Supported tag/attribute pairs

Scheduled update supports the following tag/attribute pairs when performing recursive URL updates:

✔ <a href=“ ”>

✔ <img src=“ ”>

✔ <img href=“ ”>

✔ <body background=“ ”>

✔ <frame src=“ ”>

✔ <iframe src=“ ”>

✔ <fig src=“ ”>

✔ <overlay src=“ ”>

✔ <applet code=“ ”>

✔ <script src=“ ”>

✔ <embed src=“ ”>

✔ <bgsound src=“ ”>

✔ <area href=“ ”>

✔ <base href=“ ”>

✔ <meta content=“ ”>

While scheduled update is designed to operate on URL sets consisting of hundreds of input URLs (expanded to thousands when recursive URLs are included), it is not intended to operate on massively large URL sets, such as those used by Internet crawlers, for example.

Format

Each line in the `update.config` file uses the following format:

```
URL\Request_headers\Offset_hour\Interval\Recursion_depth\
```

The following table describes each field:

Field	Allowed inputs
URL	HTTP and FTP-based URLs.
Request_headers	(Optional) A <CR><LF> separated list of headers passed in each `GET` request. You can define any request header that conforms to the HTTP specification. The default is no request header.
Offset_hour	Base hour used to derive the update periods. The range is 00-23 hours.
Interval	The interval, in seconds, at which updates should occur, starting at Offset hour.
Recursion_depth	The depth to which referenced URLs are recursively updated, starting at the given URL.

Examples

The following example illustrates an HTTP scheduled update:

```
http://www.inktomi.com\User-Agent: noname user agent\13\3600\5\
```

This example specifies the URL & Request headers, an offset hour of 13 (1 pm), an interval of one hour, and a recursion depth of 5. This would result in updates at 13:00, 14:00, 15:00, and so on. To schedule for an update to occur only once a day, use an interval value of 24 hours x 60 minutes x 60 seconds = 86400.

The following example illustrates an FTP scheduled update:

```
ftp://anonymous@ftp.inktomi.com/pub/misc/test_file.cc\\18\120\0
```

This example specifies the FTP request, an offset hour of 18 (6 pm), and an interval of every two minutes. Note that the user must be *anonymous* and the password must appear in the `records.config` file using the configuration variable `proxy.config.http.ftp.anonymous_passwd`.

winnt_intr.config

The winnt_intr.config configuration file is used for Windows only. It contains a list of network interfaces available on the PC.

Each line in the winnt_intr.config file has the following format:

```
interface_name IPaddress
```

where:

- ✔ interface_name is the name of the network interface. A default name (for example, intr0) is created automatically during installation by the installation program. If you change the interface name in this file, you must also change any variables in the records.config file that contain the interface name (for example, proxy.config.icp.icp_interface and proxy.config.cluster.ethernet_interface).
- ✔ IPaddress is the static IP address assigned to the network interface.

Specifying URL Regular Expressions (url_regex)

This section describes how to specify a url_regex. Entries of type `url_regex` within the configuration files use regular expressions to perform a match.

The following table offers examples to illustrate how to create a valid `url_regex`:

Value	Description
x	Matches the character 'x'
.	Match any character
^	Specifies beginning of line
$	Specifies end of line
[xyz]	A “character class.” In this case, the pattern matches either ‘x’, ‘y’, or ‘z’.
[abj-oZ]	A “character class” with a range. This pattern matches ‘a’, ‘b’, any letter from ‘j’ through ‘o’, or ‘Z’.
[^A-Z]	A “negated character class”. For example, this pattern matches any character except those in the class.
r*	Zero or more r’s, where r is any regular expression
r+	One or more r’s, where r is any regular expression
r?	Zero or one r’s, where r is any regular expression
r{2,5}	From two to five r’s, where r is any regular expression
r{2,}	Two or more r’s, where r is any regular expression
r{4}	Exactly 4 r’s, where r is any regular expression
"[xyz]\"foo"	The literal string [xyz]"foo
\X	If X is ‘a’, ‘b’, ‘f’, ‘n’, ‘r’, ‘t’, or ‘v’, then the ANSI-C interpretation of \x. Otherwise, a literal 'X'. This us used to escape operators such as ‘*’.
\0	A NULL character
\123	The character with octal value 123
\x2a	The character with hexadecimal value 2a
(r)	Matches an r; where r is any regular expression. You can use parentheses d to override precedence.
rs	The regular expression r, followed by the regular expression s
r|s	Either an r or an s
#<n>#	Inserts an “end” node causing regular expression matching to stop when reached. The value n is returned.

Examples

You can specify *`dest_domain`*`=inktomi.com` to match any host in `inktomi.com`. Likewise, you can specify *`dest_domain`*`=.` to match any request.

Appendix E

Event Logging Formats

This appendix contains the following sections:

- *Inktomi custom logging fields, on page 302* provides descriptions of Inktomi logging fields
- *Logging format cross reference, on page 304* provides cross-references between Inktomi logging fields, and Netscape and Squid logging fields (including Netscape Extended and Extended-2 fields)

Inktomi custom logging fields

The following table describes the Inktomi custom logging fields.

%<field symbol>	Description
{HTTP header field name}cqh	Logs the information in the requested field of the client request HTTP header; for example, `%<{Accept-Language}cqh>` logs the Accept-Language: field in client request headers.
{HTTP header field name}pqh	Logs the information in the requested field of the proxy request HTTP header; for example, `%<{Authorization}pqh>` logs the Authorization: field in proxy request headers.
{HTTP header field name}psh	Logs the information in the requested field of the proxy response HTTP header; for example, `%<{Retry-After}psh>` logs the Retry-After: field in proxy response headers.
{HTTP header field name}ssh	Logs the information in the requested field of the server response HTTP header; for example, `%<{Age}ssh>` logs the Age: field in server response headers.
caun	The client authenticated user name; result of the RFC931/ident lookup of the client user name.
cfsc	The client finish status code; specifies whether the client request to the proxy was successfully completed (FIN), or interrupted (INTR).
chi	The client host IP; the IP address of the client's host machine.
cqbl	The client request transfer length; request body length (bytes) from client to proxy.
cqhl	The client request header length; request header length (bytes) from client to proxy.
cqhm	The client request HTTP method; method (GET, POST, ...) from client to proxy (subset of cqtx).
cqhv	The client request HTTP version.
cqtd	The client request timestamp date; specifies the date of the client request in the format yyyy-mm-dd, where yyyy is the 4-digit year, mm is the 2-digit month, and dd is the 2-digit day.
cqtn	The client request timestamp; date and time of the client's request (in the Netscape timestamp format).
cqts	The client request timestamp in Squid format; date and time of the client request, in seconds since January 1, 1970.
cqtt	The client request timestamp time; specifies the time of the client request in the format hh:mm:ss, where hh is the two-digit hour in 24-hour format, mm is the two-digit minutes, and ss is the 2-digit seconds. For example, 16:01:19.
cqtx	The full HTTP client request text, minus headers; for example, `GET http://www.inktomi.com HTTP/1.0`
cqu	The client request URI; universal resource identifier (URI) of the request from client to proxy (subset of cqtx).
cquc	The client request canonical URL; differs from cqu in that blanks (and other characters that might not be parsed by log analysis tools) are replaced by escape sequences. The escape sequence is just the ASCII code number.
cqup	The client request URL path; specifies the argument portion of the URL (everything after the host). For example, if the URL is `http://www.inktomi.com/images/foo.gif`, then this field displays `/images/foo.gif`.
cqus	The client request URL scheme (HTTP, FTP, etc.).
crc	The cache result code; specifies how the cache responded to the request (HIT, MISS, ...).

%<field symbol>	Description
pfsc	The proxy finish status code; specifies whether the proxy request to the server was successfully completed (FIN), or interrupted (INTR).
phn	The proxy hostname; the hostname of the server that generated a log entry in collated log files.
phr	The proxy hierarchy route; the route that the proxy used to retrieve the document.
pqbl	The proxy request transfer length; request body length (bytes) from proxy to server.
pqhl	The proxy request header length; request header length (bytes) from proxy to server.
pqsi	The proxy request server IP (0 on cache hits, parent-ip for requests to parent proxies).
pqsn	The proxy request server name.
pscl	The proxy response transfer length; response length (bytes) from proxy to client.
psct	The proxy response content type; content type of the document (e.g., img/gif) from server response header.
pshl	The proxy response header length; response header length (bytes) from proxy to client.
psql	The proxy response transfer length in Squid format (includes header and content length).
pssc	The proxy response status code; the HTTP response status code from proxy to client.
shi	The IP address resolved from the DNS name lookup of the host in the request. For hosts with multiple IP addresses, this field will record the IP address resolved from that particular DNS lookup. This can be misleading for cached documents. For example, if the first request was a cache miss and came from IP1 for server S and the second request for server S resolved to IP2, but came from the cache, the log entry for the second request will show IP2.
shn	The server hostname; the hostname of the origin server.
sscl	The server response transfer length; response length (bytes) from server to proxy.
sshl	The server response header length; response header length (bytes) from server to proxy.
sshv	The server response HTTP version (1.0, 1.1, ...).
sssc	The server response status code; the HTTP response status code from server to proxy.
ttms	The transfer time; total transfer time in milliseconds.
ttmsf	transfer time in milliseconds as a fractional number of seconds; specifies the transfer time of the document in millisecond resolution, but instead of formatting the output as an integer (as with ttms), the display is formatted as a floating point number representing a fractional number of seconds. For example, if the transfer time is 1500 milliseconds, this field displays 1.5 while the ttms field displays 1500 and the tts field displays 1.
tts	The transfer time in seconds; specifies the transfer time of the document in seconds.

Logging format cross reference

The following sections illustrate the correspondence between Inktomi logging fields and standard logging fields for the Squid and Netscape formats.

Squid logging formats

The following table lists the Squid logging fields and the corresponding Inktomi logging field symbols:

Squid	Inktomi field symbols
time	cqts
elapsed	ttms
client	chi
action/code	crc/pssc
size	psql
method	cqhm
url	cquc
ident	caun
hierarchy/from	phr/pqsn
content	psct

For example, if you want to create a custom format called `short_sq` based on the first three Squid fields, enter a line in the `logs.config` file as follows:

```
format:enabled:1:short_sq:%<cqts> %<ttms> %<chi>:short_sq:ASCII:none
```

See *Using custom formats, on page 161* for more information about defining custom log files.

Netscape Common logging formats

The following table lists the Netscape Common logging fields and the corresponding Inktomi logging field symbols:

Netscape Common	Inktomi field symbols
host	chi
usr	caun
[time]	[cqtn]
“req”	“cqtx”
s1	pssc
c1	pscl

Netscape Extended logging formats

The following table lists the Netscape Extended logging fields and the corresponding Inktomi logging field symbols.

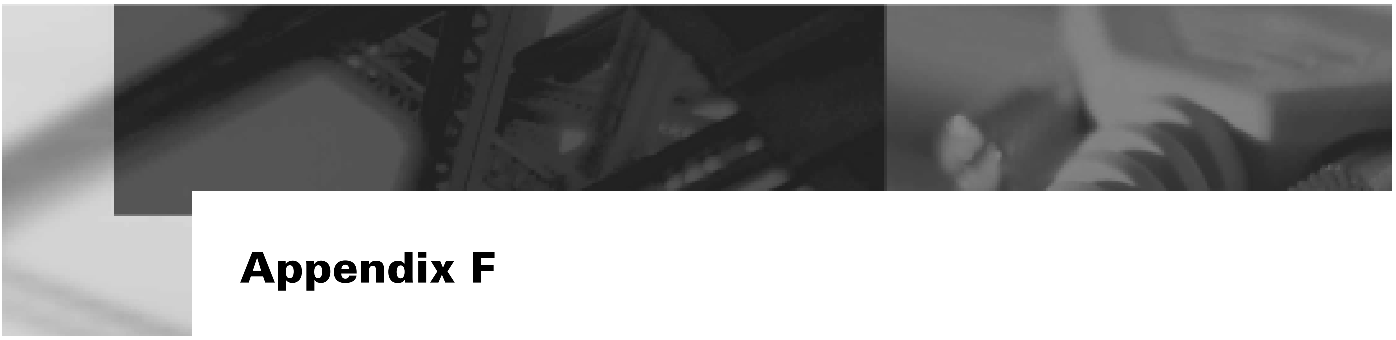
Netscape Extended	Inktomi field symbols
host	chi
usr	caun
[time]	[cqtn]
“req”	“cqtx”
s1	pssc
c1	pscl
s2	sssc
c2	sscl
b1	cqbl
b2	pqbl
h1	cqhl
h2	pshl
h3	pqhl
h4	sshl
xt	tts

Netscape Extended-2 logging formats

The following table lists the Netscape Extended-2 logging fields and the corresponding Inktomi logging field symbols.

Netscape Extended-2	Inktomi field symbols
host	chi
usr	caun
[time]	[cqtn]
“req”	“cqtx”
s1	pssc
c1	pscl
s2	sssc
c2	sscl
b1	cqbl
b2	pqbl
h1	cqhl
h2	pshl
h3	pqhl
h4	sshl
xt	tts
route	phr
pfs	cfsc

Netscape Extended-2	Inktomi field symbols
ss	pfsc
crc	crc



Appendix F

Traffic Server Error Messages

This appendix contains the following sections:

- *Traffic Server error messages, on page 308* describes the messages that Traffic Server software sends to the system log file (`syslog`) in UNIX or the Event Viewer in Windows
- *Traffic Server alarm messages, on page 310* describes the alarm messages that appear in the Traffic Manager UI Monitor pages
- *HTML messages sent to clients, on page 311* describes the HTML error messages that Traffic Server sends to browser clients (not to be confused with standard HTTP response codes)
- *Standard HTTP response messages, on page 314* describes the standard HTTP response codes that origin servers send to browser clients

Traffic Server error messages

The following table lists messages that can appear in `syslog` files (UNIX) or the Event Viewer (Windows). This list is not exhaustive; it describes messages that can occur and may require your attention.

Traffic Server Notes

Message	Description
machine down <IP address>	Machine with given IP address is down.
machine up <IP address>, protocol version=<X.Y>,	Machine with given IP address and protocol version is up.
Cluster notes	
Cluster <IP address> not in config, declaring down	
Cluster bbwrite to <IP address> failed, declaring down	
Cluster network connection to <IP address> backing up	
Cluster read from <IP address> failed, declaring down	
Cluster write to <IP address> failed, declaring down	
Illegal cluster connection from <IP address>	
machine down <IP address:port>	
machine down <IP address>	
machine up <IP address:port>	
machine up <IP address>	
Network congestion to <IP address> cleared, reverting to cache mode	Congestion is cleared and cache capability returns.
Network congestion to <IP address> encountered, reverting to proxy only mode	Traffic Server is too congested to cache and is reverting to proxy only mode.
Unable to accept cluster connections	
Host database notes	
reconfiguring host database	
<Hostname>'s round robin DNS entry updated, entries=<entry number>, IP list: <IP list name>	
Logging notes	
rolled file <file name> already exists, attempting version <version>	Attempting to roll over existing file, so roll is being changed.
"\"Vary: <header field>" —object not served from cache	Document content varies on header fields, so the cached copy is not being served back to the client.

Traffic Server Process fatal

Message	Description
accept port is not between 1 and 65535. Please check configuration	The port specified in the `records.config` file for accepting incoming HTTP requests is not valid.
self loop is detected in parent proxy configuration	The name and port of the parent proxy are the same as that of Traffic Server. This creates a loop when Traffic Server attempts to send the request to the parent proxy.

Traffic Server Warnings

Message	Description
<Log file> error: <error number>	Generic logging error.
Bad cluster major version range <version1-version2> for node <IP address> connect failed	Incompatible software versions causing a problem.
cache read error	If this message appears by itself, it might indicate a cache read problem; call Technical Support. If it appears with other cache warnings such as unable to read cache segment, marking segment corrupt, or unable to write pool header, there is a disk problem. You may have to replace your disk.
can't open config file <config file name> for reading custom formats	Custom logging was enabled, but Traffic Server cannot find the `logs.config` file.
connect by disallowed client <IP address>, closing	The specified client is not allowed to connect to the Traffic Server proxy. The client IP address is not listed in the `ip_allow.config` file.
Could not rename <log file name> to <rolled file name>	System error in renaming log file during roll.
Did <this amount> of backup still to do <remaining amount>	Congestion is approaching.
Different clustering minor versions <version 1, version 2> for node <IP address> continuing	Incompatible software versions causing a problem.
log format symbol <symbol name> not found	Custom log format references a field symbol that does not exist. See *Appendix E, Event Logging Formats*.
marking segment <segment identifier> corrupt	Traffic Server is marking the indicated part of a disk as corrupt. You may have to replace the disk.
missing field for field marker	Error in reading a log buffer.
No storage available. Cache disabled.	There is no cache storage available. There is a configuration or hardware problem. Check for other `syslog` messages that give more specific information.
Unable to accept cluster connections on port: <cluster port number>	Call technical support
Unable to open logfile <file name>, errno = <error number>	Cannot open the log file.
unable to read cache segment	The garbage collector is unable to read cache segments in the following cases: if the segment is corrupt, if the pool the segment is in is corrupt, or if there is an actual disk error while trying to read the segment. This warning is usually accompanied by a warning indicating that the segment is being marked as corrupt.

Traffic Server alarm messages

The following table describes alarm messages that you may see on the **Dashboard** page of the Traffic Manager UI.

Message	Description
[Rollback::Rollback] Config file is read-only: <file name>	Go to Traffic Server's `config` directory and check the indicated file permissions; change them if necessary.
[Rollback::Rollback] Unable to read or write config file <filename>	Go to Traffic Server's `config` directory and make sure the indicated file exists. Check its permissions and change them if necessary.
[Traffic Manager] Configuration File Update Failed: <error number>	Go to Traffic Server's `config` directory and check the indicated file permissions; change them if necessary.
[Traffic Manager] Mgmt <==>Proxy conn. closed	This is an informational message informing you that the `traffic_server` process was down. For example, you would see this message if there was a restart.
Access logging suspended - configured space allocation exhausted.	The space allocated to the event log files is full. You must either increase the space or delete some log files to enable access logging to continue. To prevent this from happening, consider rolling log files more frequently and enabling the autodelete feature. See *Rolling event log files, on page 167*.
Access logging suspended - no more space on the logging partition.	The entire partition containing the event logs is full. You must delete or move some log files to enable access logging to continue. To prevent this from happening, consider rolling log files more frequently and enabling the autodelete feature. See *Rolling event log files, on page 167*.
Created zero length place holder for config file <file name>	Go to Traffic Server's `config` directory and check the indicated file. If it is indeed zero in length, use a backup copy of the configuration file.
Traffic Server can't open <file name> for reading custom formats	Make sure that the `proxy.config.log2.config_file` variable in the `records.config` file contains the correct path to the custom log configuration file (the default is `logging/logs.config`).
Traffic Server could not open logfile <file name>	Check permissions for the indicated file and the logging directory.
Traffic Server failed to parse line <line number> of the logging config file <file name>	Check your custom log configuration file. There may be some syntax errors. See *Inktomi custom logging fields, on page 302* for correct custom log format fields.
vip_config binary is not setuid root, manager will be unable to enable virtual ip addresses	The `traffic_manager` is not able to set virtual IP addresses. You must setuid root for the `vip_config` file in Traffic Server's `bin` directory.

HTML messages sent to clients

Traffic Server returns detailed error messages to browser clients when there are problems with the HTTP transactions requested by the browser. These Traffic Server response messages correspond to standard HTTP response codes, but provide more information. A list of the more frequently encountered HTTP response codes is provided on *page 314*. You can customize Traffic Server's response messages.

The following table lists Traffic Server's hard-coded HTTP messages, their corresponding HTTP response codes, and their corresponding customizable files.

Title	HTTP code	Description	Customizable file name
Access Denied	403	You are not allowed to access the document at location *URL*.	access#denied
Bad HTTP request for FTP Object	400	Bad HTTP request for FTP object.	ftp#bad_request
Cache Read Error	500	Error reading from cache. Please retry request.	cache#read_error
Connection Timed Out	504	Server has not sent any data for too long a time.	timeout#inactivity
Content Length Required	400	Could not process this request because no Content-Length was specified.	request#no_content_length
Cycle Detected	400	Your request is prohibited because it would cause an HTTP proxy cycle.	request#cycle_detected
Forbidden	403	*port_number* is not an allowed port for SSL connections. (You have made a request for a secure SSL connection to a forbidden port number.)	access#ssl_forbidden
FTP Authentication Required	401	You need to specify a correct username and password to access the requested FTP document *URL*.	ftp#auth_required
FTP Connection Failed	502	Could not connect to the server *server_name*.	connect#failed_connect
FTP Error	502	The FTP server *server_name* returned an error. The request for document *URL* failed.	ftp#error
Host Header Required	400	An attempt was made to transparently proxy your request, but this attempt failed because your browser did not send an HTTP `Host` header. Manually configure your browser to use http://*proxy_name*:*proxy_port* as an HTTP proxy. Refer to your browser's documentation for details. Alternatively, end users can upgrade to a browser that supports the HTTP `Host` header field.	interception#no_host

Title	HTTP code	Description	Customizable file name
Host Header Required	400	Your browser did not send a *Host* HTTP header field and therefore the virtual host being requested could not be determined. To access this web site correctly, you will need to upgrade to a browser that supports the HTTP *Host* header field.	request#no_host
HTTP Version Not Supported	505	The origin server *server_name* is using an unsupported version of the HTTP protocol.	response#bad_version
Invalid HTTP Request	400	Could not process this *client_request* HTTP method> request for *URL*.	request#syntax_error
Invalid HTTP Response	502	The host *server_name* did not return the document *URL* correctly.	response#bad_response
Malformed Server Response	502	The host *server_name* did not return the document *URL* correctly.	response#bad_response
Malformed Server Response Status	502	The host *server_name* did not return the document *URL* correctly.	response#bad_response
Maximum Transaction Time exceeded	504	Too much time has passed transmitting document *URL*.	timeout#activity
No Response Header From Server	502	The host *server_name* did not return the document *URL* correctly.	response#bad_response
Not Cached	504	This document was not available in the cache, and you (the client) only accept cached copies.	cache#not_in_cache
Not Found on Accelerator	404	The request for *URL on* host *server_name* was not found. Check the location and try again.	urlrouting#no_mapping
NULL	502	The host *hostname* did not return the document *URL* correctly.	response#bad_response
Proxy Authentication Required	407	Please login with username and password.	access#proxy_auth_required
Server Hangup	502	The server *hostname* closed the connection before the transaction was completed.	connect#hangup
Temporarily Moved	302	The document you requested,*URL*, has moved to a new location. The new location is *new_URL*.	redirect#moved_temporarily
Transcoding Not Available	406	Unable to provide the document *URL* in the format requested by your browser.	transcoding#unsupported
Tunnel Connection Failed	502	Could not connect to the server *hostname*.	connect#failed_connect
Unknown Error	502	The host *hostname* did not return the document *URL* correctly.	response#bad_response

Title	HTTP code	Description	Customizable file name
Unknown Host	500	Unable to locate the server named *hostname*. The server does not have a DNS entry. Perhaps there is a misspelling in the server name, or the server no longer exists. Double-check the name and try again.	connect#dns_failed
Unsupported URL Scheme	400	Cannot perform your request for the document *URL* because the protocol scheme is unknown.	request#scheme_unsupported

Standard HTTP response messages

The following standard HTTP response messages are provided for your information. For a more complete list and descriptions, see the Hypertext Transfer Protocol — HTTP/1.1 Specification.

Message	Description
200	OK
202	Accepted
204	No Content
206	Partial Content
300	Multiple Choices
301	Moved Permanently
302	Found
303	See Other
304	Not Modified
400	Bad Request
401	Unauthorized; retry
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not acceptable
408	Request Timeout
500	Internal server error
501	Not Implemented
502	Bad Gateway
504	Gateway Timeout

Glossary

Alternates

Different versions of the same web object. Some origin servers answer requests to the same URL with a variety of objects. The content of these objects can vary widely, depending on whether a server delivers content for different languages, targets different browsers with different presentation styles, or delivers variable content at different times of the day.

ARM

Adaptive Redirection Module. Used in transparent proxy caching, ARM is a Traffic Server component that redirects intercepted client traffic destined for an origin server to the Traffic Server application. Before the traffic is redirected by the ARM, it is intercepted by an *L4 switch* or router.

Cache

Stores copies of frequently accessed objects close to users and serves them to users when requested. See also *Object store*.

Cache hierarchy

Levels of caches that communicate with each other. All cache hierarchies recognize the concepts of *Parent cache* and *Child cache*.

Cache hit

An object in the cache that can be served directly to the client.

Cache miss

An object that is *not* in the cache or that is in the cache but no longer valid. In both cases, Traffic Server must get the object from the *Origin server*.

Caching web proxy server

A web proxy server with local cache storage that allows the proxy to fulfill client requests locally, using a cached copy of the origin server's previous response.

CGI

Common Gateway Interface. A set of rules that describe how an origin server and another piece of software (a CGI *program*) located on the same machine communicate.

cgi-bin

The most common directory name on an origin server in which *CGI* programs are stored.

Child cache

A cache lower in a *Cache hierarchy* for which Traffic Server is a parent. See also *Parent cache*.

Cluster

A group of Traffic Server nodes that share configuration information and can act as a single large virtual cache.

Configure mode

One of two modes in *Traffic Manager* and *Traffic Line*. Configure mode lets you configure the Traffic Server system. See also *Monitor mode*.

Cookie

A piece of information sent by an origin server to a web browser. The browser software saves the information and sends it back to the server whenever the browser makes additional requests from the server. Cookies enable origin servers to keep track of users.

DNS

Domain Name Service. Traffic Server includes a fast, asynchronous DNS resolver to streamline conversion of hostnames to IP addresses.

Explicit proxy caching

A Traffic Server configuration option where client software (typically a browser) must be specifically configured to send web requests to the Traffic Server proxy.

FTP

File Transfer Protocol. A protocol based on TCP/IP for reliable file transfer.

Full clustering

A Traffic Server cluster distributes its cache across its nodes into a single, virtual object store, rather than replicating the cache, node by node. See also *Management-only clustering*.

HTTP

HyperText Transfer Protocol. The client-server protocol upon which the World Wide Web is based.

ICP

Internet Cache Protocol. A protocol for proxy caches to exchange information about their content.

IP

Internet Protocol. The lowest-layer protocol under TCP/IP responsible for end-to-end forwarding and long packet fragmentation control.

ISP

Internet Service Provider. An organization that provides access to the Internet.

JavaScript

A network-oriented programming language specifically designed for writing programs that can be safely downloaded to your computer through the Internet.

L4 switch

An ethernet switch that can control network traffic flow using Level 4 rules. The switch can intercept desired client protocol packets and direct them to a proxy for transparent operation.

Management-only clustering

A Traffic Server option where all nodes in a cluster automatically share configuration information. See also *Full clustering*.

MIB

Management Information Base. The set of parameters that an SNMP management station can query in the SNMP agent of a network device (for example, a router). Traffic Server supports two MIBs: MIB2 (a well-known standard MIB) and the Inktomi proprietary Traffic Server MIB, which provides more specific node and cluster information.

Monitor mode

One of two modes in *Traffic Manager* and *Traffic Line*. Monitor mode lets you view statistics about Traffic Server performance and web traffic. See also *Configure mode*.

MRTG

Multi Router Traffic Grapher. A graphing tool provided with Traffic Server that enables you to monitor Traffic Server's performance.

Netscape log format

A standard access log format. Using the Netscape log format, you can analyze Traffic Server access log files with off-the-shelf log analysis scripts. See also *Squid log format*.

News server

A web server you can access to read and post to usenet newsgroups.

NNTP

Network News Transfer Protocol. A protocol used to distribute, inquire, retrieve, and post news articles.

Object store

A custom high-speed database where Traffic Server stores all cached objects.

Origin server

The web server that contains the original copy of the requested information.

Parent cache

A cache higher up in a *Cache hierarchy*, to which Traffic Server can send requests.

Plugin

An add-on feature that provides additional functionality to Traffic Server, such as origin server blacklisting, web content filtering, authentication, and data transformation.

POP

1. Point of Presence. Usually a city or location to which a network can be connected, often with dial up phone lines.

2. Post Office Protocol. The basic protocols for addressing e-mail.

Proxy server

See *Web proxy server*.

Reverse proxy

A option that allows Traffic Server to be configured as an origin server for convenient geographical distribution of server content. Reverse proxy also offloads static content service from servers building dynamic content and provides a peak load buffer or *surge protector* for origin servers.
Sometimes referred to as *Server acceleration*.

Router

A device that handles the connection between two or more networks. Routers look at destination addresses of the packets passing through them and decide which route to send them on.

Server acceleration

See *Reverse proxy*.

SNMP

Simple Network Management Protocol. A set of standards used for communication with devices connected to a TCP/IP network. SNMP-compliant devices (agents) store information about themselves in *MIB*s and provide this information to SNMP Managers.

SOCKS

A circuit-level proxy protocol that provides a tunneling mechanism for protocols that cannot be proxied conveniently.

Squid log format

A standard access log format. Using the Squid log format, you can analyze Traffic Server event log files with off-the-shelf log analysis scripts. See also *Netscape log format*.

SSL

Secure Sockets Layer. A protocol that enables encrypted, authenticated communications across the Internet. Used mostly in communications between origin servers and web browsers.

syslog

The UNIX system logging facility.

TCP

Transmission Control Protocol. An Internet Standard transport layer protocol. TCP provides reliable end-to-end communication by using sequenced data sent by IP.

traffic_cop

A Traffic Server process that periodically monitors the health of the *traffic_server* and *traffic_manager* processes by issuing heartbeat requests to fetch synthetic web pages.

Traffic Line

Traffic Server's command-line utility that enables you to monitor performance and change configuration settings.

Traffic Manager

Traffic Server's browser-based interface consisting of a series of web pages that enable you to monitor performance and change configuration settings.

traffic_manager

A Traffic Server process and the command and control facility. `traffic_manager` is responsible for launching, monitoring, and reconfiguring the *traffic_server* process. It is also responsible for the administration UI, the proxy auto-configuration port, the statistics interface, cluster administration, and *Virtual IP failover*.

traffic_server

A Traffic Server process that is the cache processing engine of the Traffic Server product. `traffic_server` is responsible for accepting connections, processing requests, and serving documents from the *Cache* or *Origin server*.

Transparent proxy caching

A configuration option that enables Traffic Server to intercept and respond to Internet requests without requiring users to reconfigure their browser settings. It does this by intercepting traffic destined for an origin server and redirecting that traffic through the Traffic Server cache.

URL

Uniform Resource Locator. The address that defines the route to a file on the web or other Internet facility.

Virtual IP failover

An option available to clustered Traffic Servers, where Traffic Server maintains a pool of virtual IP addresses that it assigns to the nodes of a cluster. If a node fails, the remaining nodes mask the fault and take over the failed node's virtual interface.

WCCP

Web Cache Control Protocol. A protocol used by Cisco IOS-based routers to redirect traffic during transparent proxy caching.

Web proxy server

Forwards client requests to *Origin servers*. The proxy server may deny requests according to filter rules or security limitations.

Web server

A computer that provides World Wide Web services on the Internet. See also *Origin server*.

WPAD

Web Proxy Auto-Discovery. A protocol that allows clients to automatically locate a web proxy, providing the benefits of a proxy without the need for explicit client configuration.

Index

COPYRIGHT NOTICES

Portions of Traffic Server include third party technology used under license. One or more of the following notices may apply in connection with the license and use of Traffic Server.

tcl-7.4 license. This software is copyrighted by the Regents of the University of California, Sun Microsystems, Inc., and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN “AS-IS” BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

RESTRICTED RIGHTS: Use, duplication or disclosure by the government is subject to the restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software Clause as DFARS 252.227-7013 and FAR 52.227-19.

SSLeay-0.6.6 License. Copyright © 1995-1997 Eric Young (eay@mincm.oz.au). All rights reserved.

Redistribution and use in source code and binary forms, with or without modification, are permitted provided that the following conditions are met:

1.Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.

2.Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3.All advertising materials mentioning features or use of this software must display the following acknowledgement: “This product incorporates cryptographic software written by Eric Young (eay@mincom.oz.au).” The word ‘cryptographic’ can be left out if the routines from the library being used are not cryptographic related.

4.If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement “This product includes software written by Tim Hudson (tjh@mincom.oz.au)”

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

RSAREF (for MD5). Copyright © 1991-2, RSA Data Security, Inc. Created 1991. All rights reserved.License to copy and use this software is granted provided that it is identified as the “RSA Data Security, Inc. MD5 Message-Digest Algorithm” in all material mentioning or referencing this software or this function.

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

Portions of Traffic Server include technology used under license from RSA Data Security, Inc.

libdb-1.85 License. Copyright © 1990, 1993, 1994 The Regents of the University of California. All rights reserved.

Redistribution and use in source code and binary forms, with or without modification, are permitted provided that the following conditions are met:

1.Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.

2.Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3.All advertising materials mentioning features or use of this software must display the following acknowledgement: This product includes software developed by the University of California, Berkeley and its contributors.

4.Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Gateway Daemon, Release 4 license. © The Regents of the University of Michigan all rights reserved. Gate Daemon was originated and developed through release 3.0 by Cornell University and its collaborators.

Copyright notices and other restrictions as they currently appear in the GateD source files include one or more of the following:

Copyright © 1995 The Regents of the University of Michigan. All rights reserved. Gate Daemon was originated and developed through release 3.0 by Cornell University and its collaborators.

THIS SOFTWARE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

GateD is based on Kirton’s EGP, UC Berkeley's routing daemon (routed), and DCN’s HELLO routing Protocol. Development of GateD has been supported in part by the National Science Foundation.

Please forward bug fixes, enhancements and questions to the GateD mailing list: gated-bug@gated.merit.edu.

Cornell Authors: Jeffrey C. Honig, Scott W. Brim

Authors: Robert Hagens and Dan Schuh

Royalty-free licenses to redistribute UMD OSPF are available from the University of Maryland, College Park. For details contact: Office of Technology Liaison, 4312 Knox Road, University of Maryland, College Park, Maryland 20742, (301) 405-4209, (301) 314-9871 fax

This software was written by Rob Coltun. rcoltun@ni.umd.edu

gd 1.3 graphics library.

Portions copyright 1994, 1995, 1996, 1997, 1998, by Cold Spring Harbor Laboratory. Funded under Grant P41-RR02188 by the National Institutes of Health.

Portions copyright 1996, 1997, 1998, by Boutell.Com, Inc.

GIF decompression code copyright 1990, 1991, 1993, by David Koblas (koblas@netcom.com).

Non-LZW-based GIF compression code copyright 1998, by Hutchison Avenue Software Corporation (http://www.hasc.com/, info@hasc.com).

libregx package. Copyright 1992, 1993, 1994, 1997 Henry Spencer. All rights reserved. This software is not subject to any license of the American Telephone and Telegraph Company or of the Regents of the University of California.

Permission is granted to anyone to use this software for any purpose on any computer system, and to alter it and distribute it, subject to the following restrictions:

1. The author is not responsible for the consequences or use of this software, no matter how awful, even if they arise from flaws in it.

2. The origin of this software must not be misrepresented, either by explicit claim or by omission. Since few users ever read sources, credits must appear in the documentation.

3. Altered versions must be plainly marked as such, and must not be misrepresented as being the original software. Since few users ever read sources, credits must appear in the documentation.

4. This notice may not be removed or altered.

Emanate. Licensee agrees to preserve and reproduce the copyright notices contained in the Program Source and Software in the same form and location as any legend appearing on or in the original from which copies are made.

Portions of Traffic Server include Emanate software developed by SNMP Research International, Incorporated. Copying and distribution is by permission of SNMP Research International, Incorporated, and relevant third parties.

INN. Portions of Traffic Server include software developed by Rich Salz. Copyright 1991 Rich Salz. All rights reserved. Revision: 1.4

Redistribution and use in any form are permitted provided that the following restrictions are met:

1. Source distributions must retain this entire copyright notice and comment.

2. Binary distributions must include the acknowledgement "This product includes software developed by Rich Salz." in the documentation or other materials provided with the distribution. This must not be represented as an endorsement or promotion without specific prior written permission.

3. The origin of this software must not be misrepresented, either by explicit claim or by omission. Credits must appear in the source and documentation.

4. Altered versions must be plainly marked as such in the source and documentation and must not be misrepresented as being the original software.

THIS SOFTWARE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

NetFactory, Inc. Portions of Traffic Server include technology used under license from NetFactory, Inc.

IP-Filter package. Portions of Traffic Server include technology used under license from Darren Reed.

Documentation Feedback

We hope you find this book useful. If you have suggestions for improving it, please send them to **docfeedback@inktomi.com**, along with the title and date of this book. Your product experience can help us improve our documentation. Thank you.

Inktomi Technical Publications Group