COMPAQ

July 2001
155D-0701A-WWEN
Prepared by:
Industry Standard Server Division

Compaq Computer Corporation

Contents

Preface 3
Caching and Proxy Services 4
Client Acceleration 7
Forward Proxy 9
Transparent Proxy 38
Server Acceleration 63
Reverse Proxy 68
Conclusion 80
Where to Get More Information 80

TaskSmart C-Series Server Deployment Guide

Abstract: Configuring the Compaq TaskSmart C-Series server is as simple as pointing and clicking. However, the placement and operation of the TaskSmart server requires a more thorough examination of network architecture and operation. This guide answers the infrastructure and architecture questions that are not addressed by the *Compaq TaskSmart C-Series Administration Guide* and respective system reference guides. The concepts within this document are applicable to both the simplest and most complex TaskSmart C-Series server deployments.

harold.plaisance@compaq.com

Notice

Preface

The Compaq *TaskSmart*™ C-Series server provides you with the best-of-breed caching functionality and simplifies the challenges of deploying a caching solution. Integration of the comprehensive management features of the TaskSmart C-Series server ensures rapid and successful deployment in your network. This guide will address questions of deployment through an examination of caching, network architecture, and network devices that interact with the TaskSmart C-Series servers.

Four other documents complement this one:

- *Compaq TaskSmart C-Series Streaming Servers Deployment Guide* # 155C-0701A-WWEN
- *Compaq TaskSmart C-Series Servers Performance Guide* # 155E-0701A-WWEN
- *Compaq TaskSmart C-Series Servers Feature Procedures Guide* # 158D-0701A-WWEN
- *Enabling LDAP Authentication on Compaq TaskSmart C-Series Server* # 158E-0701A-WWEN

These four documents cover streaming, performance, and features. The *Compaq TaskSmart C-Series Server Features Procedures Guide* is a reference to the methods that can be used to configure and manage the Compaq TaskSmart C4000 models.

This guide has two primary audiences:

- Users and system administrators who are evaluating caching and the TaskSmart C-Series server
- Users that have purchased a TaskSmart C-Series server but have questions concerning its placement or operation

While discussion does not focus on transport protocols or Transmission Control Protocol/Internet Protocol (TCP/IP) specifically, this guide assumes a basic understanding of TCP/IP networks and wide area networks (WAN) interfacing.

This guide is devised to answer the vast majority of the questions that customers may have when trying to decide where and how to deploy the TaskSmart C-Series server. Through the description of best practices and guidelines, this guide presents the information necessary to deploy a TaskSmart C-Series server in any network.

Because the number of deployment possibilities is nearly limitless, this guide will focus on deployment strategies and effective methods of providing access to the TaskSmart services. It utilizes examples and diagrams to help the reader develop an understanding that can be applied to any network environment.

This guide incorporates a tremendous amount of information. In most cases, only some of the information is relevant. The information is divided into sections that describe deployment in forward, transparent, and reverse proxy. Each section can be read as a stand-alone document. Within each section, there are topics that may not be necessary for all deployments. Each section and each topic will have a clear introduction that explains who should read each topic and who should skip to another.

Caching and Proxy Services

Bandwidth demand of users and customers is a common problem for network administrators. Network traffic is growing at nearly an exponential rate from remote points of presence (POP) at a small Internet service provider (ISP) to large enterprise campuses. Budgets do not grow similarly. Network administrators and network architects have turned to caching as a method of handling rapidly increasing load demands with modestly growing bandwidth supply. Compaq offers the TaskSmart C-Series server to quicken and ease the resource constraints on the information technology (IT) staff.

Web caching provides two primary benefits in an Internetwork environment: acceleration of request response and reduction of redundant requests. A proxy cache will provide client acceleration when deployed as a forward or transparent proxy. User requests for cached objects will be filled more quickly. In addition, the external bandwidth is greatly reduced or even eliminated in fulfilling these requests. Web caching proxies retain data accessed from the Web so that future requests can be served from local copy. An object must be retrieved from the origin server initially, but subsequent requests may be serviced from cache. The subsequent requests from cache do not have to be from the original user. The requests can be from other users for that same object. The method of filling the request from cache results in both client acceleration and bandwidth reduction.

The use of caching also results in a reduction of server load. A reverse proxy, also known as a server accelerator, uses caching to alleviate loads on Web servers by responding to requests for redundantly requested objects. Requests for dynamic or non-cacheable objects are forwarded through the cache to the origin server. The use of this deployment produces client acceleration. Reverse proxy also adds the TCP connection management as a benefit. By allowing the TaskSmart C-Series server to handle incoming connections, the origin server is shielded from the distinct user sessions. The TaskSmart C-Series server will open only a few connections for requests not in cache.

Local delivery of Web pages between the origin server and the caching appliance takes place at the speed of the local area network (LAN). The cache appliance then delivers the data to the requesting client at a rate dependent on the end user's connection bandwidth. Depending on the speed of the end user's connection to the network, this might be a speed of one-tenth the rate that data is delivered to the cache appliance. After the page is delivered to the cache, the origin server load is reduced. Reducing the load on the Web server allows it to respond more quickly to client requests. The benefits of Web caching, whether the network is a large client base or a massive Web presence, are paramount to increasing performance within networks. The goal of this document is to alleviate concerns, answer questions, and provide guidelines that will make caching an effective solution for your network.

The first thing to consider is whether the TaskSmart C-Series server should be used as a client accelerator or a server accelerator. This is the most basic question in a TaskSmart C-Series server deployment and the answer will determine which sections and topics need to be read. Networks generally provide Web content, such as a large Web-hosting site, or are used to access content. Networks that are dominated by client workstations should use client acceleration as either a forward or a transparent proxy. Networks primarily hosting content for the Internet would mainly use server acceleration.

The rest of this section describes proxy services that will be the basis for TaskSmart C-Series server deployments in both client and server accelerators. This section is recommended for all readers because it applies to forward, transparent, and reverse proxies.

Proxy Services

Proper deployment of a TaskSmart C-Series server is governed by effective placement of the available and required proxy services. Caching is not the only benefit that the TaskSmart C-Series server can bring to a network. Because a proxy server is between an end user and the requested resource, the proxy has the opportunity to examine and act on any requests. Any action taken by the proxy to complete or deny this request is a function of a proxy service. Proxy services can be generalized into filtering, logging, access control, caching, and streaming media. Deciding how and where to deploy the TaskSmart C-Series server within a network should maximize performance while delivering all required services. To understand how to best deploy the TaskSmart C-Series server, administrators and architects should understand each of the services and their impact on network users.

Filtering

The initial release of the TaskSmart C-Series servers will not provide content filtering services. These features should be supported in a later release of the product. Proxy filtering services block users from viewing material that has been deemed inappropriate. Filtering services can differ in how they determine which content should be blocked. The most common methods of determining inappropriate content are a user-defined list of Uniform Resource Identifiers (URIs), a rating system, and a subscription to a filtering service that maintains lists of URI classifications. TaskSmart C-Series servers do support filtering based on rules of particular URIs.

In some cases, the filtering requirements can dictate the optimal location and configuration of a client accelerator. Filtering does not generally apply to reverse proxy because there is little reason to filter a local website.

Logging

In many environments, every user request is logged. The proxy logging service of the TaskSmart C-Series server can record every distinct object that is requested through the proxy. These logs can be used to measure and identify traffic patterns. The TaskSmart C-Series server has very flexible logging options. The system defaults to the Squid logging protocol. It can be configured to log onto multiple formats simultaneously. Extensible markup language (XML) and standard Web server formats are configurable. A link to a site containing Squid log file analysis tools can be found at the end of this document.

Logging can play a role in determining where a TaskSmart C-Series server should be deployed because different locations in the network will provide different insight into the originator of a request. Some network administrators require user-level resolution on all requests and others require no logging at all. Logging, however, does apply to both client and server accelerators.

Access Control

Access control services prevent specified users or network addresses from accessing through the proxy. The difference between access control services and filtering services is in what can be accessed and by whom. Access control services block access to all Internet resources for specified users. Filtering services block access for all users to specified resources. The TaskSmart C-Series server can provide user access control through lightweight directory access protocol (LDAP) support or via client IP address. If your network environment doesn't currently support LDAP and you require access control by user identity, you should consider a security protocol conversion gateway to enable your TaskSmart C-Series server to access your security system.

Caching

Caching is the primary function of the TaskSmart C-Series server. Caching services play a direct role in determining where the TaskSmart C-Series server should be placed. The services reduce the number of redundant requests to the origin servers and therefore reduce the bandwidth and time required to fulfill the requests. Proxy caching services manage local copies of frequently requested static data, reducing the bandwidth consumed by repeated requests and minimizing latency to fill those requests. The TaskSmart C-Series Model 50 server also supports splitting of live streams and caching of multimedia data, further reducing bandwidth.

Placement of caches can be leveraged to reduce the bandwidth requirements of large WANs or even the smallest of remote offices. Small ISPs, schools, and businesses benefit by reduced costs and by gaining faster performance with the use of forward proxy caches. Large Web hosting systems benefit by reduced loads via reverse proxies. Understanding the nature of the traffic over various parts of your network is a key to properly choosing the best location(s) of the cache(s). Placement should minimize potential bottlenecks from the high-speed cache to the clients. Placement should also reduce the redundant requests over expensive and typically slower WAN connections.

As LAN bandwidth speeds have skyrocketed, WAN technology options for most customers have remained unchanged. Typically, the connections that are slow and expensive are circuits that operate through technologies originally designed for telephone services. Placing the cache on the side where the fast LAN connections meet the slow WAN links is common. Placing the TaskSmart C-Series server here results in client acceleration and bandwidth reduction. Depending on the complexity and extent of your network, you may need to design a hierarchical configuration with several edge caches and potentially redundant parent caches.

Streaming Media

The TaskSmart C-Series streaming server offers support for Microsoft, Real, and QuickTime formats. This enables the latest video and audio streaming formats to be utilized in your environment while minimizing the impact of moving redundant copies of these data through your network. You can create systems that ensure minimum WAN resources are utilized to ubiquitously distribute your content.

How to Proceed

Select the appropriate section—“Client Acceleration” or “Server Acceleration”—and skip other sections and topics. “Client Acceleration” has information on both the forward proxy and the transparent proxy deployments. “Server Acceleration” has information on reverse proxy deployment. The sections can be read independently, thus users need only read the relevant topics. A single TaskSmart C-Series server may be configured as a forward, transparent, and reverse proxy all at the same time. Therefore, it may be necessary to read more than one section of this guide to answer all deployment questions.

Client Acceleration

This section of the guide provides all of the information that a system administrator or network architect will need to make an effective deployment of a TaskSmart C-Series server as a client accelerator. It covers:

- Client acceleration, such as users accessing the Web, suitable for primarily downstream traffic
- Clear distinction between forward and transparent proxy, as well as the suitable environments for both

If the network is a point of publishing, with most traffic being outbound, administrators should skip this section and proceed to "Server Acceleration."

Introduction

More users continually find their way onto the Internet, which generates more traffic. However, every new user does not necessarily create an entirely new data set. Caching takes advantage of this usage pattern by eliminating redundant requests from slow and expensive transit links. Because the information was stored locally, all of the latency and congestion of the Internet is eliminated. The network is perceived faster by the end user and the external network connection load is reduced. While this theory of caching is simple, deploying a client accelerating forward proxy can raise some complex questions.

Methods of Deployment

There are two modes of deployment for a client accelerator: forward proxy and transparent proxy. Both deployments provide improved response on frequently accessed Web data and reduce redundant Web traffic on WAN connections. They differ in the configuration necessary on the client's network topology and its potential performance. To help decide which method is most appropriate for a network, examine the differences between the two.

For the TaskSmart C-Series server to reply to a request, it must receive the request. Client requests must be directed to the TaskSmart C-Series server so that it can reply to requests and cache the data for future utilization. This directing mechanism is the primary difference between the forward proxy and transparent proxy deployments. In a forward proxy deployment, the client browser is configured to send requests to a proxy. Each client Web browser must be provided proxy information such as proxy name (or address) and proxy port. Manually walking from station to station to supply the proxy information can quickly become labor and time prohibitive. A forward proxy deployment makes this configuration necessary. Desktop management packages, such as Microsoft Systems Management Server (SMS), can automate the client configuration for a forward proxy setup. You may choose a simple method by the use of a logon script at the point of user logon to set the configuration of the client browsers.

Transparent proxy can provide the same benefits if the system management tools that can automate the client configuration are not available and the client base is too large to allow manually entering the information. This would not require any configuration at the client's location. A transparent proxy does not rely on the client browser to direct requests to the proxy. A network device will intercept hypertext transport protocol (HTTP) requests and redirect them to the TaskSmart C-Series server for caching. Any device capable of intercepting and redirecting the request will suffice for a transparent proxy. Some of the more common redirecting devices are L4 switches, L7 switches, Cisco Routers with WCCP (WCCP 2.0 supported by TaskSmart C-Series servers), or routers capable of port-based policy routing. All of these devices are able to intercept a client request and forward it to the TaskSmart C-Series server.

After the TaskSmart C-Series server receives the request, operation is identical for both the forward and transparent proxy. If the object is held in cache and known to be valid, the TaskSmart C-Series server responds promptly. If not, the request is forwarded to the origin server and the reply is both cached and sent to the client. The TaskSmart C-Series server can also issue a request via get-if-modified. This allows the TaskSmart C-Series server to confirm that an object is fresh and then send the entire object to an end user. This further reduces bandwidth while providing fresh content to end users.

If the network is in need of a client accelerator, but it is unclear whether to use a forward or a transparent proxy, consider the following factors:

- **Number of workstations**—If the number of workstations on the network is small enough that managing a manually entered proxy is possible, forward proxy may be the most cost-effective solution. When the number of workstations on the network is so large that manual configuration of the browsers is not possible or reasonable, an automated method of configuring the clients is required for a forward proxy. The alternative is to deploy the TaskSmart C-Series server as a transparent proxy when the number of workstations prohibits manually entering the information and no automation tools are available.
- **Automated configuration of the clients**—If administrators employ desktop management systems, such as SMS, the proxy configuration can be automated by pushing registry settings to the clients upon logon. This can be done regardless of the number of users. In large environments, this is the only viable method of deploying a forward proxy. Again, if the number of workstations prohibits manually entering the information and no automation tools are available, the TaskSmart C-Series server should be deployed as a transparent proxy.
- **Redirecting hardware availability**—Many routers and switches are able to redirect client requests without having to reconfigure the client browsers. While these devices carry extra costs, they enable levels of performance and fault tolerance that are not possible with a forward proxy. If this hardware is available or accessible, read both the forward and the transparent proxy sections. The detailed discussions of forward and transparent proxies will provide additional information that may help decide if forward or transparent proxy is the best choice.

Neither forward proxy nor transparent proxy is better because both deliver the same benefits. The TaskSmart C-Series server proxy benefits include high-speed caching services, scalability to meet performance needs, and the ability to be configured for high fault tolerance. The choice is based on which is easier and more cost effective to deploy in the network.

How to Proceed

- If the client configurations can be managed either manually or automatically and client acceleration is required, continue with “Forward Proxy.”
- If the client configurations cannot be managed or if the network architecture does not allow a forward proxy and client acceleration is required, skip to “Transparent Proxy.”
- If it is still unclear whether forward proxy or transparent proxy is the best client accelerator, read both “Forward Proxy” and “Transparent Proxy.”
- If the network is a point of publishing, with most traffic being outbound, skip this section and proceed to “Server Acceleration.”

Forward Proxy

This section of the guide provides all of the information that a system administrator or network architect will need to make an effective deployment of a TaskSmart C-Series server as a forward proxy. This section must assume that the network is either small enough that manually configuring the clients is possible or that an automation tool is present. The term “forward” implies that every client browser has been configured to use a proxy to access the Internet. Clients will then actively *forward* all requests to the proxy.

Introduction

Client acceleration as a forward proxy requires that the client browser actively direct all requests to the proxy. To have the browser redirect the request, the connection properties of the browser must be changed at each workstation. While this configuration is simple, forward proxy may be labor-prohibitive in larger environments with hundreds or thousands of clients. SMS provides a mechanism to seamlessly automate this configuration by “pushing” a proxy configuration to clients with minimal effort.

Client acceleration as a forward proxy is somewhat easier to understand and architect because it requires no special hardware or routing configuration. This is not to say that load balancers or intelligent switches cannot be used to augment a forward proxy deployment. The only obstacle to forward proxy may be the configuration required on the clients. This drawback is often outweighed by the cost of redirecting hardware. For smaller networks or large networks with SMS, the cost of manually or automatically maintaining the client configuration might be cheaper than the hardware. A forward proxy should be used only in an environment where configuration of the client browser is possible and the clients on the network cannot use any other method of accessing the Internet.

A forward proxy would not be a suitable solution in some environments. For example, an ISP may not want to deploy a forward proxy for dial-up customers as it would:

- Require manual configuration of the client browser, which would lead to countless support calls.
- Interfere with other ISP accounts that the end user may have, as one provider’s proxy would not be available on another network.

If a forward proxy does not have a firewall or access control device preventing non-proxied access, the clients may circumvent the proxy by changing their proxy configuration.

The position and operation should maximize performance while delivering the required proxy services when deployed as a forward proxy. For the caching service, there are two main goals. The first is improving network response to the clients and the second is saving bandwidth. Other goals may be associated with additional proxy services. In many cases, other proxy services may play a role in sizing and placing a TaskSmart C-Series server. The other proxy services that relate to forward proxy are logging and access control.

Directing Requests to the Forward Proxy

The key to a forward proxy is the client browser configuration. Each Web browser maintains a proxy configuration that can be viewed in the browser settings or options. Table 1 shows how to navigate to the proxy configuration in some popular Web browsers. For streaming deployment, refer to the *Compaq TaskSmart C-Series Streaming Servers Deployment Guide* document number 155C-0701A-WWEN

Table 1. Navigate to the Proxy Configuration

Netscape Navigator	Microsoft Internet Explorer 4.x	Microsoft Internet Explorer 5.x
1. Click **Edit**.	1. Click **View**.	1. Click **Tools**.
2. Click **Preferences**.	2. Click **Options**.	2. Click **Internet Options**.
3. Expand the **Advanced** options at the bottom of the list by clicking the **+**.	3. Click **Connection** tab.	3. Click **Connection** tab.
4. Click **Proxy**.		4. Click **LAN Settings**.

When the network is small enough, it is simple to manually enter the proxy information at each of the client stations. SMS can provide a more powerful solution if the clients on the network are Microsoft Win32-based. SMS can force values for certain registry keys or send a configuration file to a client at logon.

The following uses a registry monitor, REGMON.EXE, to manually change the proxy configuration. It shows the registry settings that must to be pushed to clients using Microsoft Internet Explorer 5.0.

- HKEY_CURRENT_CONFIG\Software\Microsoft\windows\CurrentVersion\Internet Settings\ProxyEnable
- HKEY_CURRENT_USER\Software\Microsoft\windows\CurrentVersion\Internet Settings\ProxyEnable
- HKEY_CURRENT_USER \Software\Microsoft\windows\CurrentVersion\Internet Settings\ProxyServer
- HKEY_CURRENT_USER \Software\Microsoft\windows\CurrentVersion\Internet Settings\ProxyOverride
- HKEY_CURRENT_USER \Software\Microsoft\windows\CurrentVersion\Internet Settings\AutoConfigURL
- HKEY_CURRENT_CONFIG\Software\Microsoft\windows\CurrentVersion\Internet Settings\ProxyEnable

Netscape Navigator does not use the Windows registry to store proxy information. The proxy settings for a user can be found in the user's subdirectory where Netscape was originally installed. A file called PREFS.JS contains the proxy settings for a given user. By replacing this file with one containing the required proxy settings, Netscape can be forced to use the specified proxy. It may be necessary to provide both solutions, depending on the mix of browsers in the network.The settings for the proxy in this file are as follows:

- user_pref("network.proxy.http", "*x.x.x.x*");
- user_pref("network.proxy.http_port", 8080);
- user_pref("network.proxy.type", 1);

Forward Proxy Operation

Client browsers no longer have any responsibility for filling the Web request when configured to use a proxy for Internet access. The TaskSmart C-Series server performs the domain name system (DNS) and HTTP requests on behalf of the clients. The tasked Web browsers simply forward the requests to the TaskSmart C-Series server and wait for the server to retrieve the objects.

After the client has forwarded the request to the TaskSmart C-Series server, the server is responsible for the DNS resolution and HTTP requests necessary to retrieve the requested objects. While filling the client request, the TaskSmart C-Series server has the opportunity to cache all of these objects and DNS lookups. Should the TaskSmart C-Series server receive a request for an object that is not already in cache, the TaskSmart C-Series server will issue requests to the origin Web server. To the origin server the TaskSmart C-Series server seems to be just a client issuing requests. The Web server responds with the requested data and sends it to the TaskSmart C-Series server. The TaskSmart C-Series server forwards the requested data to the client and stores a copy of the object locally. Should the TaskSmart C-Series server receive a request for the same object, the server can respond with the local copy from cache. This can be accomplished with little or no interaction or delay from the origin server.

As a forward proxy, the TaskSmart C-Series server will respond to HTTP and file transfer protocol (FTP) requests on the specified proxy port, generally 8080. The TaskSmart C-Series Model 50 server supports Real Networks, Windows Media, and QuickTime streaming formats as well.

The requests filled from cache, called *cache hits*, are sent to the clients at LAN speeds. This not only eliminates the transmission delay, but also eliminates delay caused by congestion on the WAN link. The cache hits are served much more quickly than cache misses to the clients.

There are minimal changes noticeable to the end user other than the improved response. Noticeable changes from the client point of view would be in the format of the error messages. With the TaskSmart C-Series server as the proxy, any HTTP errors will be reported to the client via the TaskSmart C-Series server error page. If an end user requests a nonexistent domain, the TaskSmart C-Series server will attempt to retrieve the nonexistent domain from the Internet. The TaskSmart C-Series server will respond with an error page as a response to the end user's request, after a timeout.

Cacheable Data

TaskSmart C-Series Models 30 and 40 servers, when deployed as a forward proxy, will respond only to port 8080, proxy HTTP, and FTP requests. The TaskSmart C-Series Model 50 server streaming unit also supports Real Networks, Windows Media, QuickTime protocols for both live and on-demand requests. Requests that reach the TaskSmart C-Series server are simply forwarded to the default gateway in the configuration. Subsequent live stream requests are split while on-demand requests are cached.

Since the introduction of HTTP, there has been the ability to control intermediary caches from the origin server. Originally, this was done with the hypertext markup language (HTML) header "PROGMA NO-CACHE." HTTP 1.1 extended the definitions of cacheable to include expirations and limited cacheable data. Additional information on cacheable objects can be found in the *Compaq TaskSmart C-Series Administration Guide*, Chapter 3.

Three types of objects are not cached:

- Objects that are marked non-cacheable in the HTTP header
- Objects that exceed the maximum object size and are a configurable variable on the TaskSmart C-Series server
- Objects that are implied as non-cacheable

Table 2. Cacheable Data

Type of Data	Is it Cached?
HTTP	Yes, unless marked otherwise by origin server or exceeds maximum file size for cacheable objects.
HTTPS	No
Password-protected FTP	Yes, but only served from cache to authenticated user.
Anonymous FTP	Yes, unless it exceeds maximum file size for cacheable objects.
Password-protected FTP	No
.ASP	No, requires server-side processing. Static-embedded objects are cached.
.CGI	No, requires server-side processing. Static-embedded objects are cached.
Cookies	No, origin server should mark cookies as "PRIVATE."

An example of a non-cacheable HTML object is a document that is marked as non-cacheable or a document that implies that it is non-cacheable. Implied non-cacheable data is any HTML that requires server-side processing, such as .CGI or .ASP. This does not mean that the TaskSmart C-Series server will not accelerate .CGI or .ASP pages. TaskSmart C-Series server caching is done at the object level, not the page level. This means that all static images (.JPG, .BMP, .GIF, and so on) are cached. In a given .ASP HTML document, there may be many static cacheable objects. The static objects, such as a corporate logo, are the same every time the page is viewed. These objects will be cached and they will be served from the high-speed local cache when they are next requested as part of a dynamically generated HTML document. In terms of bandwidth, the embedded objects and images that are static and cacheable represent a considerable and large percentage of the total size of a Web page. There is a large portion of data, within dynamic pages, that is cacheable.

Another type of data that is not cached is hypertext transport protocol secure (HTTPS) or secure data. In fact, HTTPS data is simply passed through the TaskSmart C-Series server. Any banking transactions or online purchases are essentially transparent to the TaskSmart C-Series server. Since HTTPS uses port 443, the proxy services are not applied to secure sessions. Secure data should not be confused with private or password-protected data. Data that is static and requires a user name and password to be accessed is cached. If a user name and password are required, the objects are marked so that only the authenticated user can view them from the cache. Therefore, they are cached only for the single user. If another user uses a different logon to view the same information, the information will not be served from cache. The user name and password are not cached and the authentication is still performed on the origin server. The origin server generally marks cookies as non-cacheable or "private." Cookies should always be marked private if they contain user-specific data. Any objects that are marked "private" cannot be held in any cache other than the client browser.

In the HTTP header, an origin server may specify an expiration time for an object. If a redundant request is made and the expiration time has passed, the TaskSmart C-Series server will request the object via a "get-if-modified." This type of request also contains the time and date the original copy was retrieved. The server can then respond with a terse "Not Modified" response. The proxy will respond to the client with either an entire copy of the response or "Not Modified" if the client made a get-if-modified request to the proxy. The TaskSmart C-Series server is not allowed (by the HTTP specifications) to vend out objects that it believes may be expired. This means that if an object has expired and the origin server is not available, the TaskSmart C-Series server will not display the expired data.

Architecture and Placement in a Network

Undoubtedly, the most common question concerning deployment of a TaskSmart C-Series server is "Where do I put it in my network?" Three basic locations define the most effective placement of a TaskSmart C-Series server in a particular network. Examples of placement within a network are displayed in Figure 1.

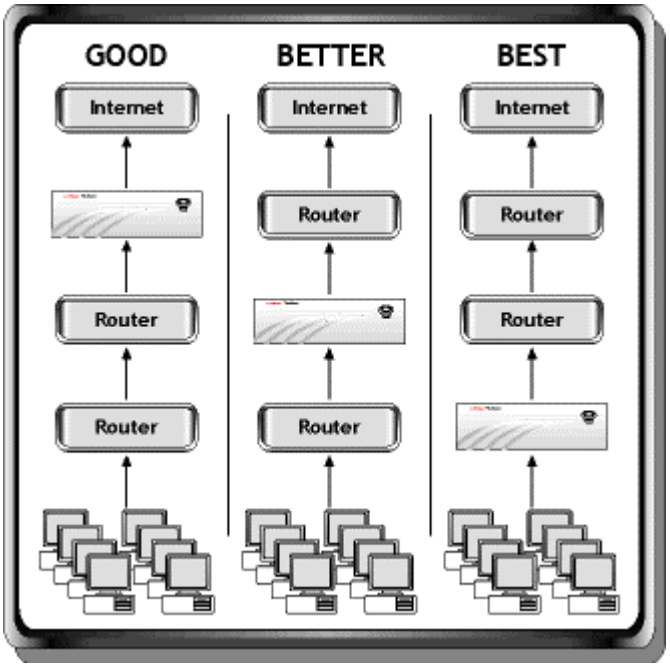

Figure 1: Minimizing hops and placing close to clients

- Place TaskSmart server logically between the clients and the Internet. Clients will submit requests to the TaskSmart C-Series server and the server will respond with a response it has received from the Internet. Placing the TaskSmart C-Series server between the client and the Internet also reduces redundant traffic on segments that connect to the Internet.
- Place the TaskSmart server as close to the clients as possible. Requests should be sent *upstream* to the TaskSmart server. The number of hops from the workstation to the Internet should be minimized. This architecture reduces latency and eliminates potential points of failure. After this practice is established and understood, the architect should then try to reduce the number of hops between the client and the TaskSmart C-Series server. As the cache hit ratio increases, more objects will be served from cache. There will be a greater volume of traffic between the end users and the TaskSmart C-Series server than between the TaskSmart C-Series server and the origin servers. Essentially, this minimizes potential bottlenecks between the TaskSmart C-Series server and the Internet.
- Place the TaskSmart server as close to the clients, but far enough away to incorporate the most users as possible to ensure the greatest return on the investment for a TaskSmart server. Network design will mandate that several hops lie between the TaskSmart server and a client base that is large enough to generate a substantial amount of Web requests.

This is acceptable unless a particular segment becomes a bottleneck. New switching network technologies allow gigabit speeds on corporate backbones and even WANs. These products may allow a more centralized approach to your configuration without impacting the performance of the system.

Generally, the locations in the network that meet these requirements are points where users are funneled together within the network (*aggregation point*) or transferred to another network (*access point*). Placing the TaskSmart C-Series server at a location where users are funneled allows the deployment to best match the server's rated load. Because there may be more than one of these aggregation points in a network, the TaskSmart C-Series server should be deployed at the point that best matches its rated load. Deploying the TaskSmart C-Series server at an access point maximizes the number of potential users of a single larger cache as depicted in Figure 2.

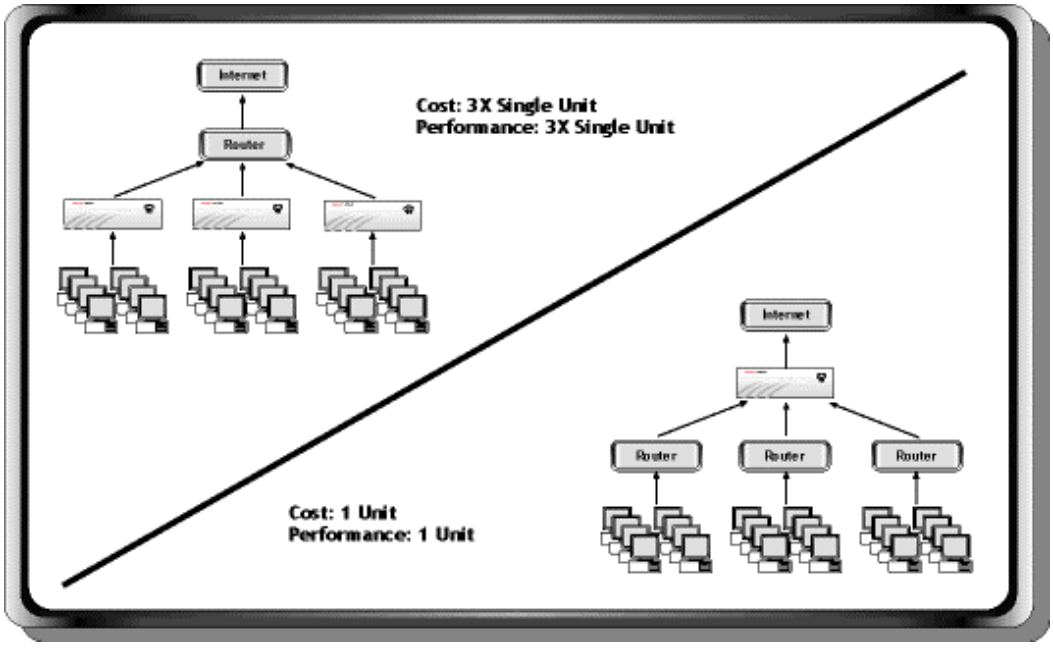

Figure 2: Matching loads to placement

Physical Connections

With the option of a gigabit interface and the standard dual 100-Megabit Ethernet interfaces on a single TaskSmart C-Series server, questions may remain concerning implementation of the architecture through physical connections. A requirement of the physical connections is that they must provide stable connections to all necessary network resources. The TaskSmart C-Series server must be able to reach the default gateway for proper operation. The clients, servers, and DNS services must be accessible, directly or through a network gateway, on one or more of the physical connections.

While all models of the TaskSmart C-Series server have at least two Ethernet Network Interface Cards (NICs), this does not imply that the appliance must be in-line with the Ethernet switch or hub that connects the end users to the network. Most applications will be achieved by a single connection between the appliance and the Ethernet switch where end users or common network servers are connected to the network. The physical interfaces should provide the maximum bandwidth required to serve the clients. The use of different interfaces for each subnet or scale of connection to a gigabit interface may be required for this level of support. Streaming media can generate rates in the hundreds of megabits. Large switched LAN architectures may require the gigabit option to handle the traffic without congestion. All objects that are cached must first be retrieved from the origin server. Be sure to consider peek load conditions of the traffic necessary to fulfill the origin request and the load conditions of client response.

Forward Proxy TaskSmart Hierarchies

This section provides all of the information needed to make an effective deployment of a forward proxy TaskSmart C-Series server hierarchy. It assumes that the network is large enough to require multiple levels of caching and that the TaskSmart C-Series server will be used at every level. “Forward Multi-Proxy Hierarchies” explains a heterogeneous proxy hierarchy, where servers other than TaskSmart C-Series server will be members of the hierarchy. If the network is small with only one proxy necessary to meet all performance and proxy service requirements, skip to “Other Network Devices.”

Up to this point, all proxy configurations that were considered were single-level caches. TaskSmart C-Series servers can be deployed in a very powerful, layered architecture called a hierarchy. As the name implies, a hierarchy has multiple smaller caches that are grouped below a larger cache.

Consider a large corporation with many remote offices. Each remote office may have a WAN connection back to the home office via a frame relay network. Large WANs created in a star topology may best benefit from a cache placed as close as possible to the center of the WAN network. An example would be fractional T3 connections to a frame relay network, where many offices are connected via T1s. If the cache were placed at the same point where the frame relay network connects to the T3, the cache could serve the remote offices without requiring redundant data to move all the way to the home office. This might be a good location for a pair of parent proxies that serve edge caches at each of the remote offices. This configuration would enable a single copy of a high bandwidth video stream to be served to all the nodes in a large enterprise WAN. This can be done without overloading the interconnecting WAN links. Figure 3 provides a detailed campus scenario.

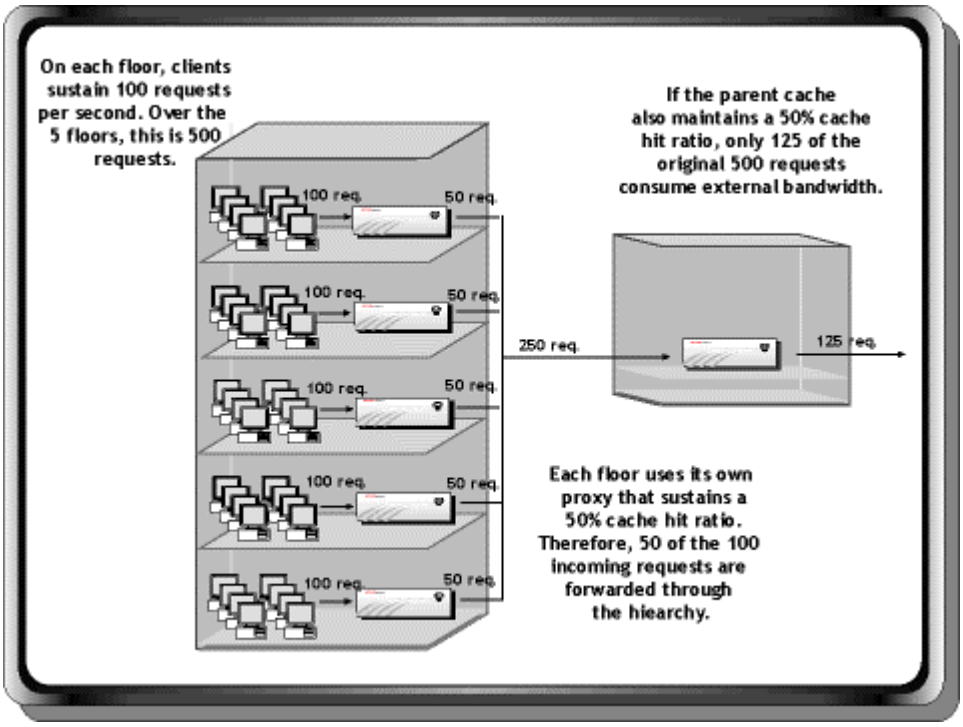

Figure 3: Hierarchy benefit and operation

The bandwidth savings of a TaskSmart C-Series server hierarchy can be tremendous in large environments. Modern networking technologies have scaled bandwidth to a point where most networks can handle Web browsing without such an extensive deployment. Video is quite a different scenario. Most corporate personal computers (PCs) can now handle high bandwidth video. Networks, however, become overwhelmed from the bandwidth requirements of these new communication systems.

TaskSmart C-Series servers issue requests to peers and parents via the Internet Cache Protocol (ICP) or HTTP cache hierarchy. ICP is a widely accepted protocol and is supported by nearly all of the caching proxies. This means that cache hierarchies can be built with other brands of caching solutions, even Squid freeware. For more information on the TaskSmart C-Series server deployment with other proxies, see “Forward Multi-Proxy Hierarchies.” In this topic, hierarchies will consist only of TaskSmart C-Series servers.

Hierarchy Relationships

Alone, a single-level TaskSmart C-Series server deployment reduces redundant traffic generated by users. As a hierarchy, groups of TaskSmart C-Series servers can not only reduce redundant traffic for users, but also reduce the redundant traffic among *groups* of users if each group is connected to its own proxy. The bandwidth savings of a TaskSmart C-Series server hierarchy can be substantially higher than a single-level deployment in the same environment.

In a hierarchy, TaskSmart C-Series servers that are nearer to the clients are called *children*. The servers that are one level up in the hierarchy are *parents*. In a three level cache, there are no grandparents. Instead, the parents of the first hierarchy are now considered the *children* in a new hierarchy.

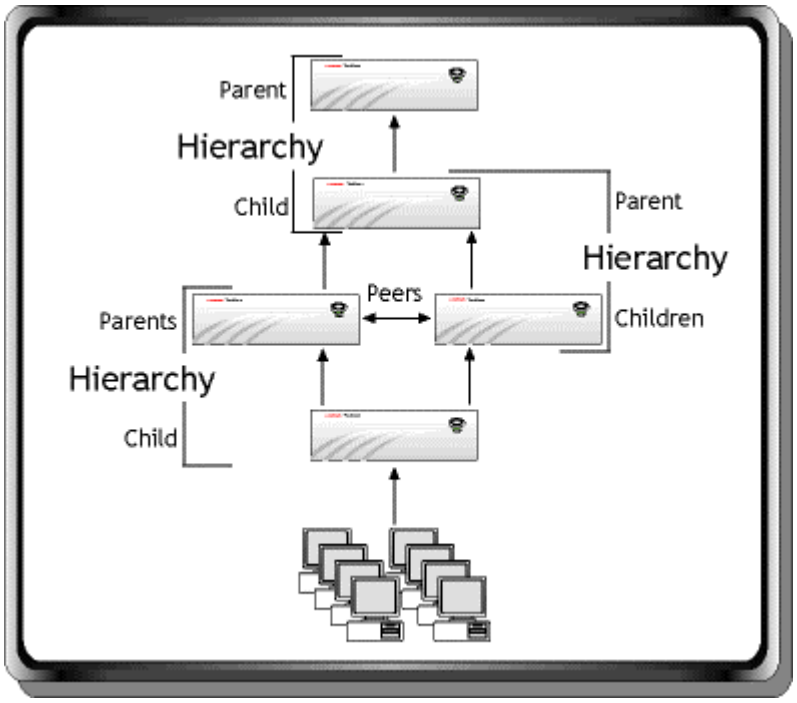

Figure 4: Parents and peers in multilevel hierarchy

While a network may have multiple levels, the parent and child are the only two logical levels in any hierarchy. Each hierarchy is considered independent of levels above or below it. This means that in a hierarchy, even if there are four levels, only two of them are listed in any single hierarchy configuration.

In hierarchies consisting exclusively of TaskSmart C-Series servers, there are only three possible relationships:

- ICP Parent
- ICP Child
- ICP Peer

ICP Parent

When deployed as an ICP parent, the TaskSmart C-Series server will respond to requests that are submitted from proxies that are closer to the clients. An ICP parent does not need to know the addresses of the child caches below it. The children must be aware of the parent's address and the parent must know only that it is a parent. As an ICP parent, a TaskSmart C-Series server would have one or more caches from which it receives requests when the child caches could not be filled. In some cases, the ICP parent will be required to fill the request on behalf of the client. In this situation, the requested object would then be held in both the child and parent caches for future use.

ICP Child

As an ICP child, the TaskSmart C-Series server will forward all requests to an upstream parent cache before attempting to fill the request. An ICP child can also be configured to fill all requests through the parent, even if the parent does not already have the object in cache. The configuration necessary for an ICP child is simply specifying an ICP parent. The ICP child is not an explicit hierarchy entity.

ICP Peer

ICP-peered caches are similar to the relationship between children and parents, except that peers cannot be configured to fill ICP misses through their peers. If an ICP peer requests an object of a peer and receives an ICP "miss" reply, the requesting peer must either fill the request itself or forward the request to any specified parents. ICP peering is not necessarily bi-directional. TaskSmart C-Series server A may have TaskSmart C-Series server B as a peer, but TaskSmart C-Series server B does not need to have TaskSmart C-Series server A as a peer.

Architecture and Placement in a Hierarchy

Unlike the concept itself, the configuration for a hierarchy is very simple. On a TaskSmart C-Series server, the configuration for both ICP and HTTP Proxy is the same. Simply indicate the addresses of the parents and any peers in the hierarchy if using ICP.

Note: European Organization for Nuclear Research's server software (CERN) is synonymous with the HTTP Proxy. HTTP Proxy is used in this document and in the software for the TaskSmart C-Series server but the functionality is the same.

Positioning parents in relationship to children can be considered the same way placement is decided with a single TaskSmart C-Series server. There are three primary requirements for the relative placement of members of a hierarchy.

- The parent server should be placed logically between the children and the Internet. Any requests that the child cache attempts to fill from a parent should travel towards the Internet and the origin server. Placing a parent cache farther within the network will create additional internal traffic and delays to filling the client request.
- Place the parent server as close to the children as possible. Reducing hops and distance between members of a hierarchy will reduce latency between them. Keep in mind that the concept and performance of a hierarchy implies a large distributed network. As a result, there may be several hops between siblings. Try to reduce unnecessary hops.
- Place the parent server close to the parent. By placing the parent near the children, the redundant traffic through the LAN and to the WAN will be minimized. A hierarchy should be sized such that the parent will be able to handle all requests that cannot be filled from cache. If there is any traffic that does not originate from the children, this must be taken into account when sizing the TaskSmart C-Series server.

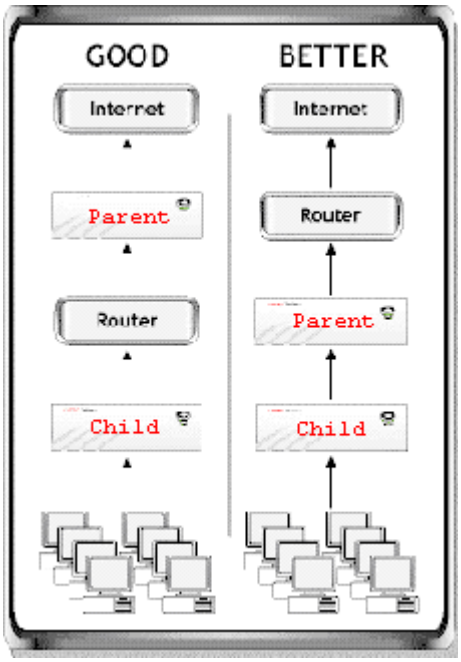

Figure 5: Reducing hops between parents and children

TaskSmart Proxy Services in a Hierarchy

This section details deployment options and considerations for networks that will use only TaskSmart C-Series servers to provide the proxy services in the hierarchy. Because TaskSmart C-Series servers can provide the same proxy services, the relative placement of the services will be determined more as a matter of performance or convenience.

Authentication

The TaskSmart C-Series servers provide LDAP user authentication if required. All forms of plain text authentication will work with a transparent TaskSmart C-Series server hierarchy. HTTPS operating through secure sockets layer (SSL) can be proxied by the TaskSmart C-Series server. If Windows NT Challenge/Response (NTLM) authentication is used on the Web server, then no member of the hierarchy should proxy the client request to that server. Instead, the browser should be configured to bypass the proxy for requests bound for an NTLM authenticating server.

Access

The TaskSmart C-Series servers support IP address and port-based restriction. The TaskSmart C-Series server can restrict services based on IP address. If IP-based access control is used, be sure that no Network Address Translation (NAT) equipment can obfuscate the end user's IP address.

Logging

Logging requires that the log service be deployed at the level that allows the proper resolution. Applying the logging service as a parent allows logging of the child proxy requests only. For logging at the user-level, the logging service must be applied at the proxy closest to the clients. Logging at the parent level in a hierarchy would show only the children issuing requests, unless another group of users is accessing the parent directly. The TaskSmart C-Series server has very flexible logging options. It will default to the Squid logging protocol. The TaskSmart C-Series server can be configured to log on multiple formats simultaneously. XML and standard Web server formats are configurable. A link to a site containing many Squid log file analysis tools can be found at the end of this document.

Caching

Caching is different from the other proxy services. Caching is the only one that can benefit by having multiple layers of redundant service. Therefore, if TaskSmart C-Series servers are deployed in a hierarchy, use all available services to create a caching hierarchy that will increase the bandwidth savings.

Sizing a Hierarchy

A hierarchy should be sized such that the parent should be able to handle all requests that cannot be filled by all children. Because each hierarchy is only aware of two levels, there is no practical limit on the number of levels to the architecture. A four-level architecture of TaskSmart C-Series servers is more than capable of handling the requests for tens of thousands of users. Therefore, it is unlikely that a network would ever have more than four levels in the proxy hierarchical architecture. After the hierarchy is configured, the cache-monitoring tabs on the graphical user interface can provide details on the number of requests that were received through the hierarchy using ICP or HTTP Parent Caching.

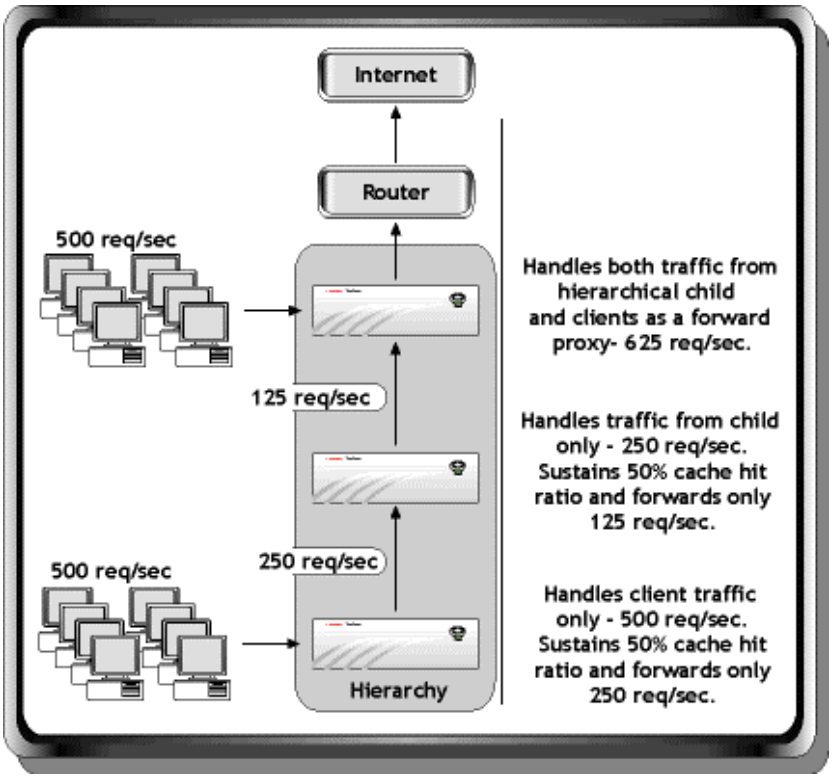

Figure 6: Sizing the hierarchy to handle client and child traffic

Forward Multi-Proxy Hierarchies

This topic will attempt to provide all of the information needed to make an effective deployment of a forward proxy hierarchy. The methods and architecture for deploying TaskSmart C-Series servers as part of a hierarchy with other proxy servers are explained. It is assumed that the network is large enough to require the multiple levels of caching and that TaskSmart C-Series servers will not be used at every level. If the network is small with only one proxy necessary to meet all performance and proxy service requirements or if you are considering a network in which TaskSmart C-Series servers will provide all the proxy services and access control is not needed, skip to "Other Network Devices."

When building hierarchies, it will often be necessary to include a proxy server other than a TaskSmart C-Series server. This could be a Microsoft Proxy server, BorderManager, or Squid. There are a few things to consider when building hierarchies with servers other than TaskSmart C-Series servers. There are several methods for a successful deployment; one is no better than the other. Whichever method has the greatest return on the investment for a given environment is the correct method of deployment. The return can be measured by performance, cost, or a combination of both. It is important to understand the goals and expected return. Perhaps the first aspect to understand is the services that the current proxy is providing. When there is a clear understanding of the current proxy's role, the optimal location for the TaskSmart C-Series server is easier to determine.

Heterogeneous Hierarchical Relationships

When building a hierarchy, proper operation and optimal performance will rely on the correct protocols used to forward requests between the members of the hierarchy. Just as with TaskSmart C-Series server hierarchies, a proxy hierarchy consists of children and parents. Logically, children are closer to the clients than they are to the parents. Operationally, children submit requests to parents but parents do not submit requests to the children.

Every type of hierarchy must have a method of forwarding requests. The TaskSmart C-Series server can use either ICP or HTTP parent caching to forward requests to parents. A TaskSmart C-Series server can receive requests on the standard proxy (port 8080) interface from an ICP child or from an HTTP child. Because ICP is an open standard supported by nearly all proxy caches, the ICP protocol will provide the greatest flexibility for future growth when building a hierarchy. The most notable exception to the list of proxies that does not support ICP is Microsoft Proxy 2.0. Instead, the TaskSmart C-Series server will use the HTTP cache protocol to pass requests to a Microsoft Proxy server as a parent.

There are only four possible hierarchical configurations for TaskSmart C-Series servers:

- ICP Parent
- ICP Child
- ICP Peer
- HTTP Cache Hierarchy

ICP Parent

When deployed as an ICP parent, the TaskSmart C-Series server will respond to requests that are submitted from proxies that are closer to the clients. An ICP parent does not need to know the addresses of the child caches below it. The children must be aware of the parent's address and the parent must know only that it is a parent. As an ICP parent, a TaskSmart C-Series server would have one or more caches from which it receives requests when the child caches could not be filled. In some cases, the ICP parent will be required to fill the request on behalf of the client. In this situation, the requested object would then be held in both the child and parent caches for future use.

ICP Child

As an ICP child, the TaskSmart C-Series server will forward all requests to an upstream parent cache before attempting to fill the request. An ICP child can also be configured to fill all requests through the parent, even if the parent does not already have the object in cache. The configuration necessary for an ICP child is simply specifying an ICP parent. The ICP child is not an explicit hierarchy entity.

ICP Peer

ICP-peered caches are similar to the relationship between children and parents, except that peers cannot be configured to fill ICP misses through their peers. If an ICP peer requests an object of a peer and receives an ICP "miss" reply, the requesting peer must either fill the request itself or forward the request to any specified parents. ICP peering is not necessarily bi-directional. TaskSmart C-Series server A may have TaskSmart C-Series server B as a peer, but TaskSmart C-Series server B does not need to have TaskSmart C-Series server A as a peer.

HTTP Cache Hierarchy

When deployed as a "parent" to a Microsoft Proxy server, the Microsoft Proxy server must have the TaskSmart C-Series server listed as an HTTP Cache Hierarchy. The TaskSmart C-Series server is in other words an "upstream Web proxy." This configuration is possible only on Microsoft Proxy. When configured to use an HTTP Cache Hierarchy, Microsoft Proxy will submit requests that it cannot fill locally to the TaskSmart C-Series server. To the HTTP Cache Hierarchy, these requests are formatted just as a proxied client request. For a TaskSmart C-Series server to function as an HTTP Cache parent, the forward proxy service must be enabled.

Protocols in a Multi-Proxy Hierarchy

ICP Hierarchies

For an ICP hierarchy, child servers need only know the ICP address and port for the parent servers. Squid, Cisco, BorderManager, Infolibria, CacheFlow, and many others support ICP. For configuring each of these servers to use ICP within a hierarchy, refer to the appropriate server documentation.

Architecturally and operationally, an ICP hierarchy of different types of proxy caches is identical to an ICP hierarchy of TaskSmart C-Series servers. The ICP protocol is a standard that should not differ from one proxy to another.

HTTP Cache Hierarchies

TaskSmart C-Series servers often need to be deployed along with a non-ICP compliant proxy. The most notable configuration is the Microsoft Proxy and TaskSmart C-Series server hierarchy. Based on the discussion of the proxy services, it should be clear whether the TaskSmart C-Series server should be between the clients and the Microsoft Proxy server or between the Microsoft Proxy server and the Internet. Unlike ICP hierarchies, the configurations for the different placements are quite different.

If the TaskSmart C-Series server is between the Microsoft Proxy server and the Internet, as the case would be for user-level access control, then the TaskSmart C-Series server must be configured as a standard forward proxy with no special hierarchical configuration. Microsoft Proxy requires an upstream Web proxy to forward requests to another proxy. In this architecture, the Microsoft Proxy server may still control user access to the Internet. All authorized users will pass through the TaskSmart C-Series server where other proxy services can be applied to authorized users.

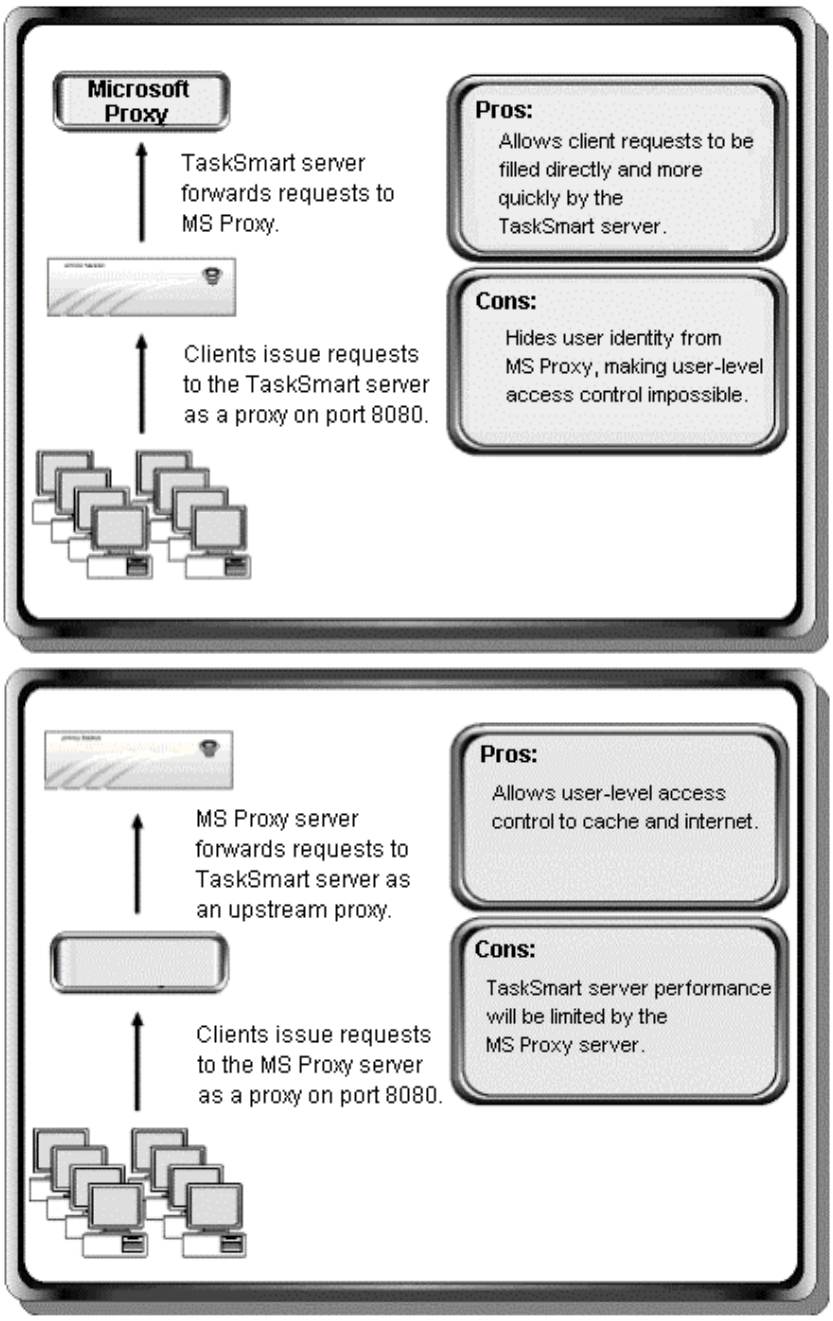

Figure 7: Placement options with the TaskSmart C-Series server and Microsoft Proxy

Note that the TaskSmart C-Series server, in this configuration, can see only the Microsoft Proxy server and not the clients. Therefore, any logging of client access would have to be done by the Microsoft Proxy server. Another item to note in this configuration is that the performance of the TaskSmart C-Series server will be limited to that of the Microsoft Proxy server for cache "misses" and fresh cache checks.

For the other scenarios, where user-level resolution is not necessary, reversing the logical positions of the Microsoft Proxy server and the TaskSmart C-Series server may be required. This would allow the Microsoft Proxy server to provide system-level (as opposed to user-level) access control, filtering, caching, and logging.

The Microsoft Proxy server requires no extra configuration beyond a standard forward proxy configuration. On the TaskSmart C-Series server, there must be a setup of a hierarchy analogous to the upstream Web proxy as required in the previous configuration. To forward the requests to the Microsoft Proxy server, configure the TaskSmart C-Series server to consider the Microsoft Proxy server as an HTTP parent. The Microsoft Proxy server configuration is unique, as Microsoft Proxy 2.0 does not support the ICP protocol. A host of other proxies and caches support ICP, providing the ability to architect overlap within the caching framework.

Architecture and Placement in a Multi-Proxy Hierarchy

When building hierarchies of different types of proxies, the architecture cannot be reduced to simple guidelines. In most cases, the reason for the mixed hierarchy will be to deliver a proxy service that one proxy does not offer. For access control, filtering, and logging, the relative placement of the proxies may enhance or inhibit the delivery of proxy services. This section discusses each proxy service and the implications of the relative placements.

Common Placement for all Proxy Hierarchy

Place the TaskSmart C-Series server as close to the clients as possible. Just as before, placing the TaskSmart C-Series server closer to the clients eliminates latency on the response to the client, reduces propagation of redundant traffic through the network, and eliminates potential bottlenecks between the TaskSmart C-Series server and the clients. Also place the TaskSmart C-Series server as far from the clients as performance and services require. As discussed in the following sections, some proxy service requirements will dictate the most effective location for the TaskSmart C-Series server.

Hierarchy Architecture with an Access Control Proxy

The TaskSmart C-Series server should be as close to the clients as possible. The easiest placement to consider is dictated by access control. So, quite easily, the distribution of access control services is decided. The relative placement of the servers is dictated by the access control requirements. While an effective deployment minimizes the distance between the clients and TaskSmart C-Series server, this may not always lead to the proper application of proxy services. If user-level access control is required, then the user access control proxy must be placed closest to the clients. This is the only arrangement that allows the access control service to see the identity of the client. If group-level or network-level access control is needed, deploying the controlling proxy upstream will provide the necessary control.

To enforce access control on a user, a proxy must have some visibility of the user identity. This can be done either by IP address or by accessing a user database, such as a domain or directory. The type of authentication can directly influence the relative placement of the TaskSmart C-Series server and the other proxy in the hierarchy.

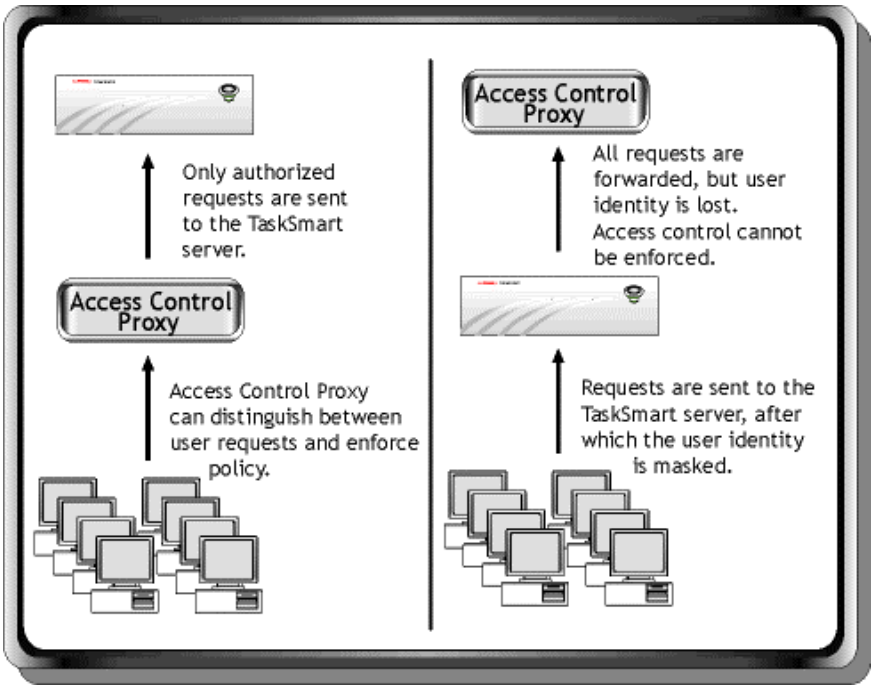

Figure 8: Effect of access control on placement

For plain-text authentication methods, where the authenticating server requests user name and password, the TaskSmart C-Series server can forward the authentication request from the authenticating server to the client. The authenticating server can then compare the credentials against a local database, the domain information, or a policy stored in a directory. If the access control server verifies that the client may access the requested information, the TaskSmart C-Series server would then be authorized to fill the request on behalf of the user. Note that if the requested objects were cacheable, they would then be cached only for the authenticated user. Other users who may or may not have access to the information will still be prompted for user name and password before the TaskSmart C-Series server would be allowed to fill the request. As a result, plain-text authentication will not directly influence the relative placement of the TaskSmart C-Series server because it will work properly despite the placement relationship to the other proxy. When using the Windows NT Challenge/Response (NTLM) authentication, the TaskSmart C-Series server cannot sit between the client and the authenticating server. NTLM does not support authentication through a proxy. Therefore, if NTLM will be used, the Microsoft Proxy server must be placed closer to the clients than the TaskSmart C-Series server. Similarly, IP-based authentication requires that the TaskSmart C-Series server proxy communicate directly to the controlled clients. One configuration is not better than another. The correct configuration is the one that allows the proper delivery of proxy services.

Hierarchy Architecture with a Logging or Filtering Proxy

Logging also requires that the log service be deployed at the level that allows the proper resolution. Applying the logging service as a parent allows only logging of the child proxy requests. A parent in a hierarchy cannot distinguish between clients on the other side of its own children. For logging at the user level, the logging service must be applied at the proxy closest to the clients.

Likewise, filtering can be deployed at different points in the network to enable either user-level or group-level filtering. If all users are subject to the same filters, it does not matter which level of the hierarchy is handling the filtering service. If all requests are forced through the hierarchy, the filtering service is being applied to all users.

Practicality may force one level or another to handle the filtering service because of filtering methods. Proxy filtering services block users from viewing material that has been deemed inappropriate. Filtering services can differ in how they determine which content should be blocked. The most common methods of determining inappropriate content are a user-defined list of Uniform Resource Identifiers (URIs), a rating system, and a subscription to a filtering service that maintains lists of URI classifications.

For all proxy services to work properly in a forward proxy environment, the firewall must be configured to not accept requests directly from the clients. If the firewall is not blocking client access, it is simple for a client to circumvent all proxy services simply by changing their browser configuration. Following these guidelines will create an architecture that makes the most effective use of the TaskSmart C-Series server high-speed cache and simultaneously meet all of the filtering requirements.

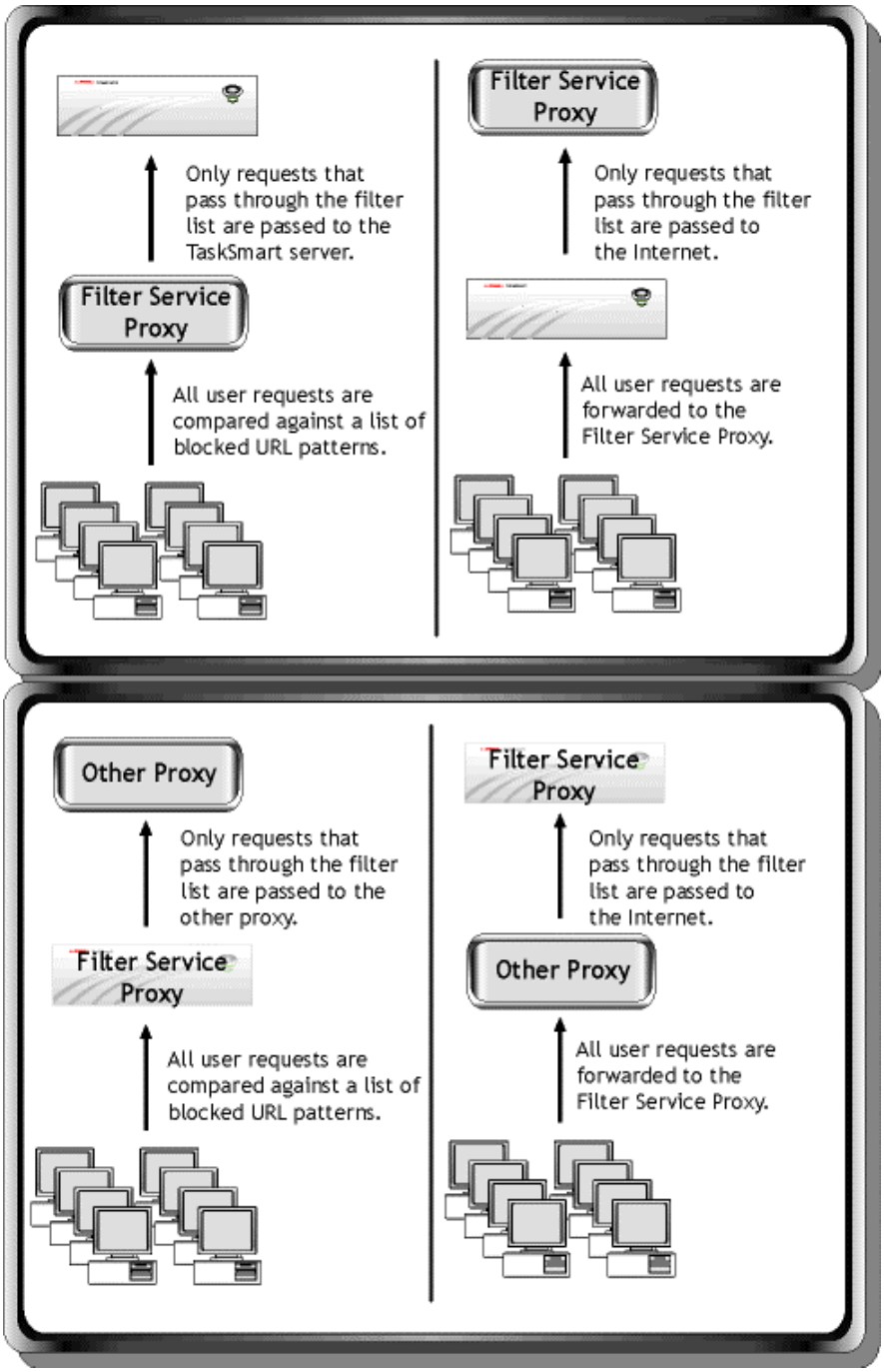

Figure 9: Placing the proxy filter service in a hierarchy

Hierarchy Architecture with a Caching Proxy

Caching is different from the other proxy services. Caching is the only proxy that can benefit by having multiple layers of redundant service. If the TaskSmart C-Series server is deployed in a hierarchy with another proxy that has its own caching service, use both services to create a caching hierarchy similar to the caching hierarchies of TaskSmart C-Series servers.

Performance rather than proxy service requirements will dictate the relative positioning of the TaskSmart C-Series server and another server providing caching services. Because the TaskSmart C-Series server is a single-function device that has been tuned for the proxy caching service, it will outperform most other proxy packages. There is the potential for the other proxy to become a bottleneck to the higher speed TaskSmart C-Series server caching service if another proxy is deployed between the clients and the TaskSmart C-Series server. Performance would be greatest if the TaskSmart C-Series server were closer to the clients than the other proxy.

Sizing a Multi-Proxy Hierarchy

A multi-proxy hierarchy should be built and sized just like a single-server deployment. Make sure that each TaskSmart C-Series server or other proxy can effectively handle all incoming requests from both the clients on the network and any children in the hierarchy. If there is any traffic that does not originate from the children, this must be taken into account when sizing the TaskSmart C-Series server.

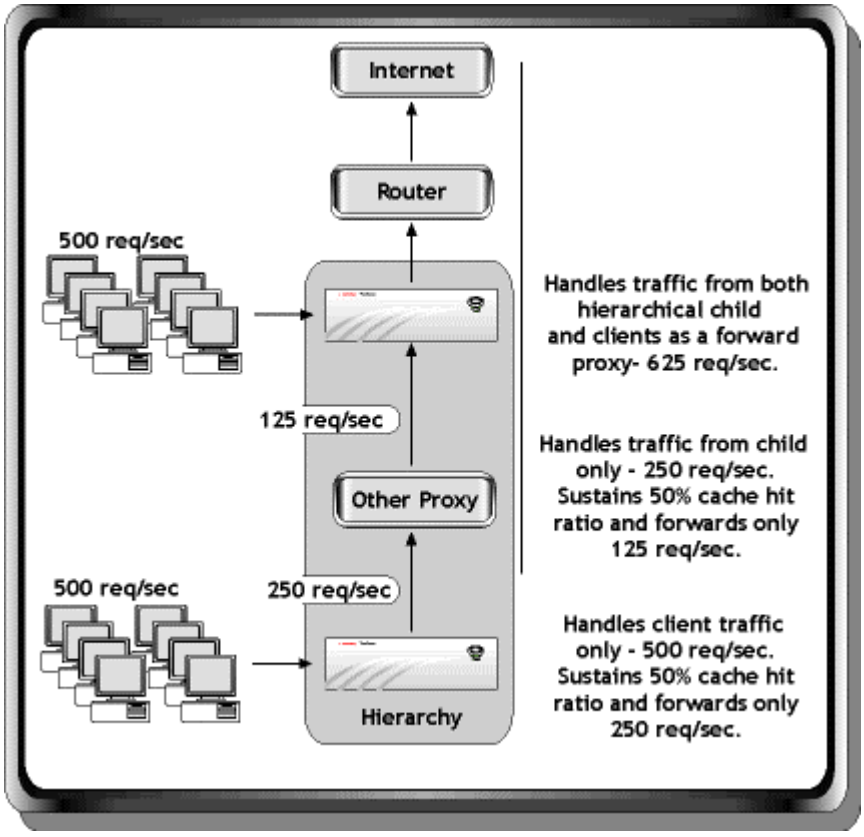

Figure 10: Sizing the hierarchy

To maximize the request rate to the TaskSmart C-Series server, administrators will need an understanding of traffic patterns at aggregation points in the network. Ideally, the TaskSmart C-Series servers should be deployed at the largest available aggregation points. This would provide the greatest probability of request overlap. For example, if a network were composed of many small user groups with each having its own Squid proxy, the point of the largest request rate would be between the Squid proxies and the Internet. If the same client base shares a single proxy, the TaskSmart C-Series server should sit between the clients and the Squid proxy as a child to the Squid proxy. In both cases, the position would place the TaskSmart C-Series server as close to the clients as possible while maximizing the incoming request rate to the TaskSmart C-Series server.

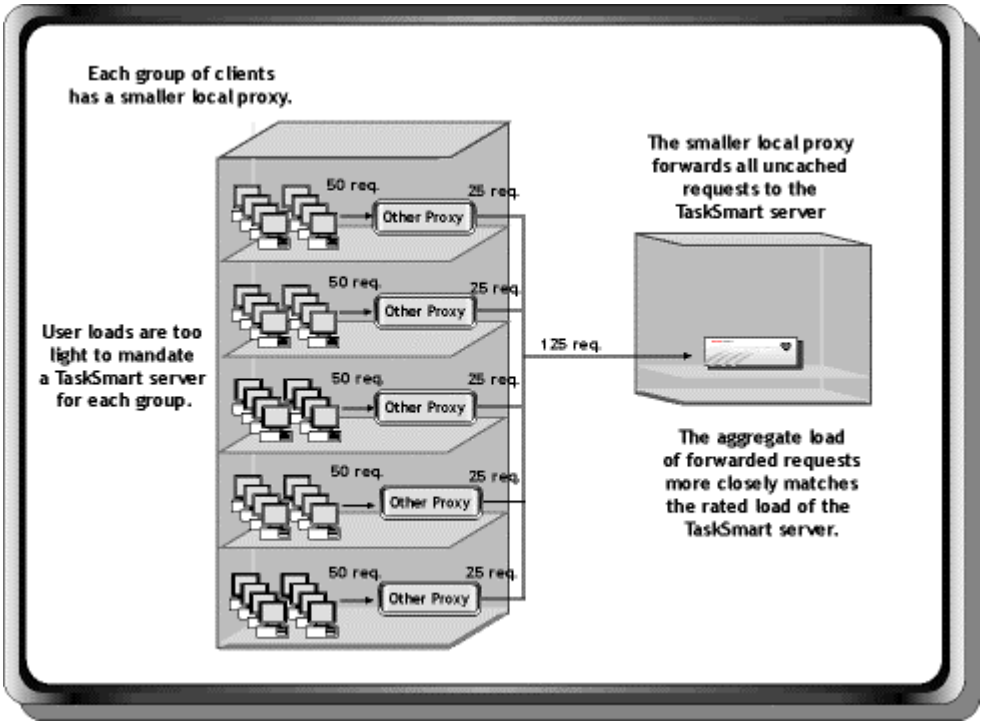

Figure 11: Mixed hierarchy with sizing considerations

A common configuration issue with hierarchies arises when the firewall allows only the parent to perform external DNS. By default, the TaskSmart C-Series server will perform its own DNS resolution and forward the requests with DNS already resolved to the parent proxy server. If the firewall is set to allow only the parent server to access the external network, then the TaskSmart C-Series server will not be able to resolve any DNS lookups that require external DNS. To force the TaskSmart C-Series server to send its requests through the parent proxy, enable the **CONFIG proxy.config.http.no_dns_just_forward_to_parent 1** option in the **records.config** file. The *Compaq TaskSmart C-Series Administration Guide* covers this in depth. In doing so, the TaskSmart C-Series server will fill all requests through its parents in the hierarchy. Be sure to also set the parent proxy in the **Traffic Server** configuration utility administrator. Enter the name or IP address followed by a colon and the port number of the parent proxy.

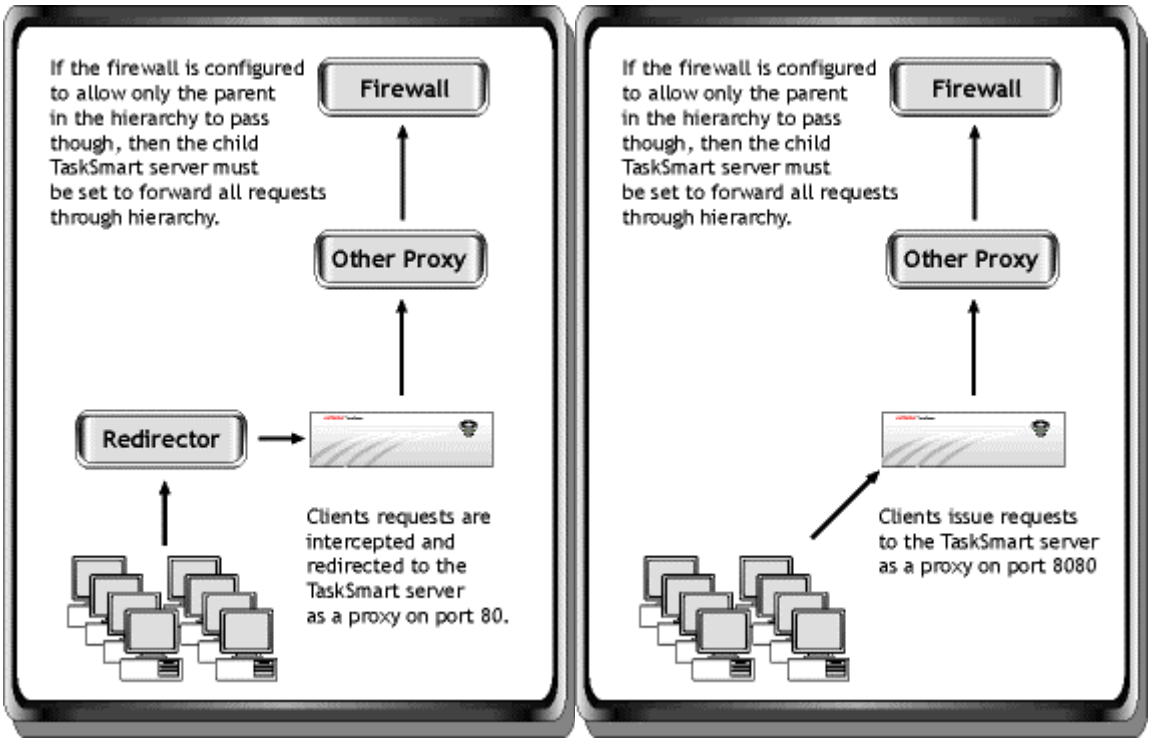

Figure 12: "Must only forward through hierarchy" scenario

Generally, there is no reason for caches in the network to have to climb the hierarchy for Intranet access or local content. If the TaskSmart C-Series server is forced to fill only through the hierarchy, DNS requests for local machines will fail because the DNS requests are being forced to the external DNS service. To correct this, TaskSmart C-Series servers allow the user to enter domains, addresses, or URLs that can be accessed directly by bypassing the proxy hierarchy.

The Microsoft Proxy server configuration is unique, as Microsoft Proxy 2.0 does not support the ICP protocol. A host of other proxies and caches support ICP, providing the ability to architect overlap within the caching framework. Making efficient use of hierarchies can drastically increase bandwidth savings in larger networks.

Firewalls and DNS Servers

This topic provides the information needed to configure other network devices to effectively deploy the TaskSmart C-Series server as a forward proxy. For a forward proxy, the only device that will have a significant impact on proxy services is a firewall. The recommended configuration for client acceleration as a forward proxy, for networks that use a firewall to restrict access, is covered. If the network does not have a firewall, skip to "Performance of a Forward Proxy."

For a forward proxy, TaskSmart C-Series server deployment, there should be no extra configuration necessary on any routers, switches, or hubs. Following best practices of minimizing latency and reducing hops between clients and the Internet is all that is necessary to make the most effective deployment of the TaskSmart C-Series server.

The network firewall must be configured to allow only the TaskSmart C-Series server or the parent of the TaskSmart C-Series server to access information directly from the Internet. This must be done for proxy filtering, logging, and access control services to work effectively when the TaskSmart C-Series server is deployed as a forward proxy. Otherwise, clients could circumvent the proxy filtering service by changing the configuration in the browser. Even if the configuration of the clients is automated, the users can still change the proxy configuration.

Additionally, Compaq recommends that the TaskSmart C-Series server be deployed within the protection of the firewall. Having the TaskSmart C-Series server within the firewall not only protects the unit from any attacks, but also allows the TaskSmart C-Series server to be closer to the clients. Additionally, if the TaskSmart server is outside of the firewall and configured as a forward proxy, anyone on the Internet can make use of the high-speed cache of the TaskSmart. In this situation, a TaskSmart server could quickly become overloaded and deliver degraded performance to the intended users.

There are two options for guaranteeing TaskSmart C-Series server access through the firewall. One is to use a parent server that can access through the firewall and have the TaskSmart C-Series server fill requests by enabling the "must only forward through hierarchy." To do this, the DNS setting must be disabled and then parent caching must be turned on at the configuration window. Detailed steps on how to do this can be found in the *Compaq TaskSmart C-Series Administration Guide*. (Refer to "Configuring traffic server to use an ICP cache hierarchy" in Chapter 7, "Hierarchical Caching." Also, references to the configuration variables can be found in Appendix D.)

The second option is to configure the firewall to allow the TaskSmart C-Series server to access resources, such as DNS, directly. If the firewall denies access to the TaskSmart C-Series server, this will generally result in a "502: Gateway timeout" error on the client.

Performance of a Forward Proxy

This topic provides all of the performance information and considerations that could impact a deployment. All administrators should read this section to better understand the performance expectations. The *Compaq TaskSmart C-Series Servers Performance Guide*, document number 155E-0701A-WWEN, covers performance levels of the TaskSmart C-Series server in depth.

Because network variables can vary from network to network, performance will be different in each network. Higher cache hit ratios with smaller objects can increase the performance significantly. Request rates and response times are both affected by many variables, such as the number of objects per page, number of users, user bandwidth, the amount of time that a user views a page, latency and load on the origin server, cache hit ratio, and size of requested objects. With so many variables, it is difficult to predict how a TaskSmart C-Series server will respond in every environment.

Determining the correct size or model of a TaskSmart C-Series server for a network can be a complex process. We can simplify the selection by limiting the variables that we consider in making the selection. The only factor that needs to be known is the request rate, in order to choose the correct model. The request rate is the number of requests that the clients issue in a given second. The current request rate can be determined by analyzing the logs of an existing proxy. If there are no existing proxies, the request rate can be approximated using a simplified formula:

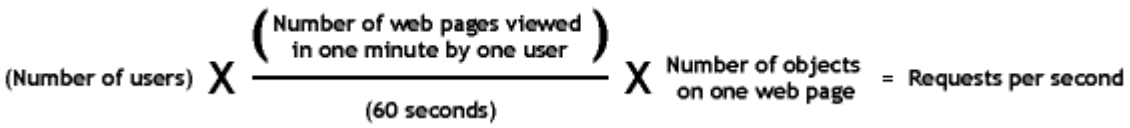

Figure 13: Request rate formula

Measuring the Effectiveness of a Forward Proxy

Because users generally do not call to report that the network is fast, positive feedback about the acceleration provided by the cache may not come readily. Some monitoring tools are able to measure response time and total retrieval time on a Web page. Administrators that are curious about the impact of the acceleration can install a response-time measuring tool. For network-wide monitoring of the impact of the caching service, a robust package such as VitalSuite (www.vitalsigns.com) will install an agent on each client and report the response time to a master server. The network-monitoring packages are generally reserved for larger enterprises.

To measure the effectiveness of bandwidth savings, the cache-hit ratio provides insight to the bandwidth saved by the cache. For example, a cache that is maintaining a 28 percent cache-hit ratio is filling 28 percent of the requests without consuming any bandwidth. The cache-monitoring panels in the TaskSmart C-Series server interface can provide the actual number of bytes that are saved by the cache. If a provider's charges are based on throughput, savings from the cache can easily be calculated.

Load Balancing and Load Distribution

This topic provides methods and mechanisms for increasing TaskSmart C-Series server performance with load balancing or load distribution. If a single TaskSmart C-Series server is not able to provide the performance necessary in the network, read this topic to learn what options are available for load balancing or load distributing forward proxy requests. If a single TaskSmart C-Series server can handle the load of the network, skip to "Fault Tolerance."

Load Balancing

The performance of a TaskSmart C-Series server can be scaled nearly linearly by using an external load-balancing device. This can be done when the service request rate of the network exceeds the rated performance of the TaskSmart C-Series servers. It also can be done if the network traffic has grown to exceed what the current TaskSmart C-Series server deployment can effectively service. Two load-balanced TaskSmart C-Series servers can handle nearly twice the number of requests per second. Up to eight units, scaled using a Foundry ServerIron switch, have been tested in Compaq labs. This configuration produced over 90 percent scalability for load-balanced TaskSmart C-Series server performance. It should be emphasized that this performance scalability requires an external load-balancing device. The TaskSmart C-Series servers cannot distribute requests between themselves. Even the TaskSmart C-Series server clustering will not enable any level of load balancing between servers.

Many load-balancing and intelligent switches can provide additional performance by load balancing requests between TaskSmart C-Series servers. As mentioned, load balancing provides nearly ideal performance scaling. Other performance-related benefits can be realized by using a load-balancing device. Many of the load-balancing devices are also capable of connection or request limiting. While limiting connections or requests will not increase performance directly, these limits will provide insurance against overloading the TaskSmart C-Series servers. Rather than allow the TaskSmart C-Series server to become overrun and refuse connections, the load-balancing device may be capable of redirecting these connections to another TaskSmart C-Series server.

Load Distribution

Using DNS round robin on the name of the proxy will distribute the loads between the multiple proxy servers. This is not a form of load balancing; it is load distribution. DNS round robin will not dynamically respond to an imbalance in load-between the proxies. DNS round robin does not handle a downed server well. A downed server in a DNS round robin configuration requires that the browser time out. After the timeout, the round robin will continue and automatically try the second server. This timeout period is rather long and noticeable to the clients. However, DNS round robin is a cost-effective method of distributing loads if no dedicated solution is available.

Another load distribution solution is to create a JavaScript proxy configuration file that parses the URL name and uses a particular proxy based on an attribute of the URL. For example, using the domain name length. This may require some basic scripting knowledge but is inexpensive and can distribute loads evenly. This solution also handles proxy failure by specifying a backup proxy if the first does not respond. For unbalanced proxy loads, there is no mechanism that can respond. Performance scaling on load distributions will vary. The more evenly loaded the servers are, the better the scaling will be.

Fault Tolerance

This topic provides methods and mechanisms for increasing TaskSmart C-Series server fault tolerance with TaskSmart C-Series server clustering and external devices. If the firewall were restricting external access for all but the TaskSmart C-Series server, client access through the firewall would fail if the TaskSmart C-Series server were unavailable. If the use of fault tolerance is the best deployment for a network, fault tolerance should be understood. If budgets or requirements do not allow fault tolerance, then skip to “Security.”

Most TaskSmart C-Series servers share features that make components fault tolerant. All TaskSmart C-Series servers have a system level fault-tolerance clustering mechanism built into the software. The software also includes a Virtual IP feature. Clustered TaskSmart C-Series servers do not increase performance without an external load-balancing device, even if they are clustered. The clustering provides system-level fault-tolerance of all proxy services. Should one member of a cluster fail or stop accepting requests, the other members of the cluster will assume the addresses and the responsibility for the proxy services on that address.

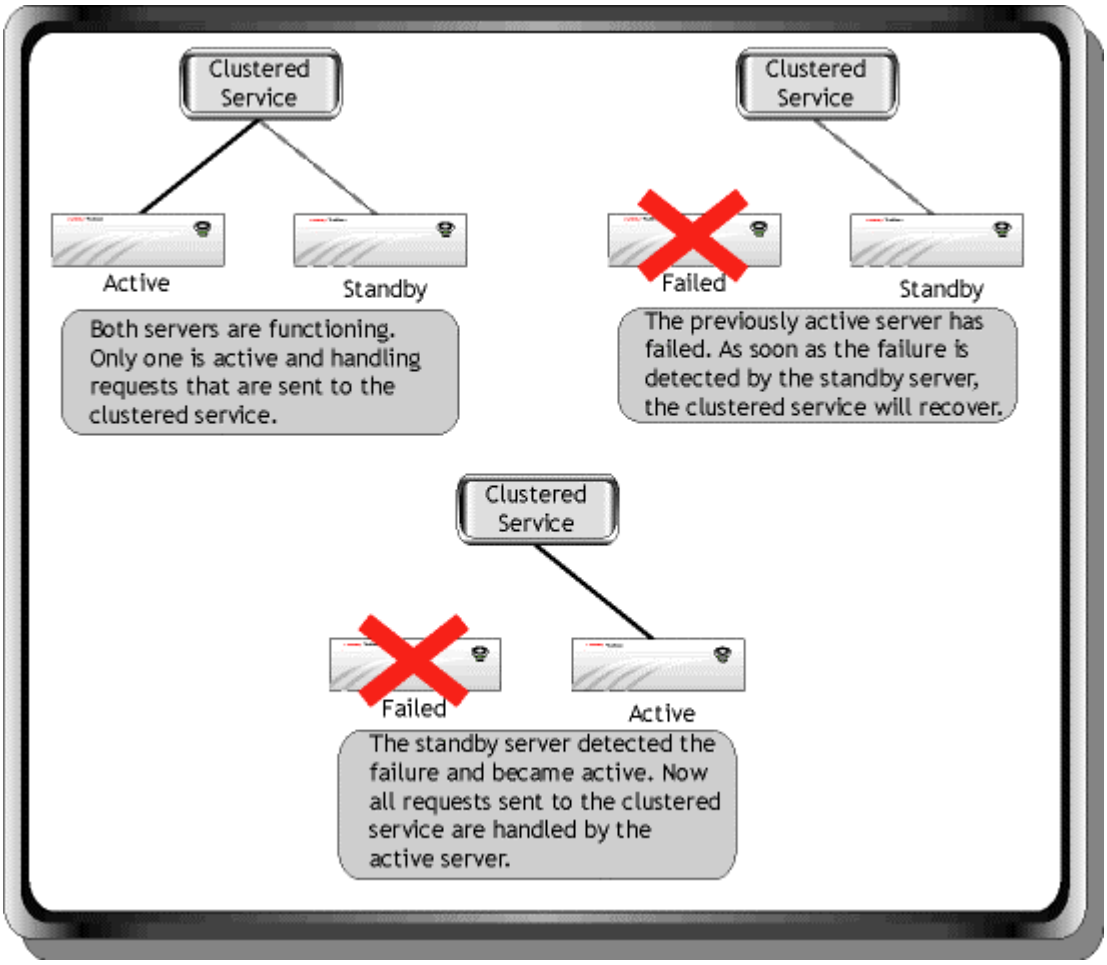

Figure 14: TaskSmart cluster failover

In any environment where the TaskSmart C-Series server is required for client connectivity, clustering should be seriously considered. Such is the case where the firewall has been configured to grant access only to the TaskSmart C-Series server. In this case, if the proxy were unavailable, clients would not be able to access any resources that lie outside the firewall. Deploying a cluster will ensure that the proxy services are fault tolerant.

By relying on an external device instead of or in addition to the clustering, the system-level fault tolerance can be enhanced beyond the simple failover of the software clustering. In the event of a failure, an intelligent switch or load-balancing device could be configured to redirect requests to another TaskSmart C-Series server. The TaskSmart C-Series server can pass the requests without redirecting or even redirect only a portion of the requests. This method of fault tolerance is the most responsive and most configurable.

The highest performance from a cluster requires an external load balancer. To distribute the loads between the TaskSmart C-Series servers in a cluster, multiple virtual addresses must be created. The load balancer can then distribute the loads evenly between the virtual addresses. In the following configuration, notice the number of virtual services. TaskSmart C-Series servers will distribute the virtual services evenly between all members of the cluster. Consider this responsibility distribution rather than load distribution.

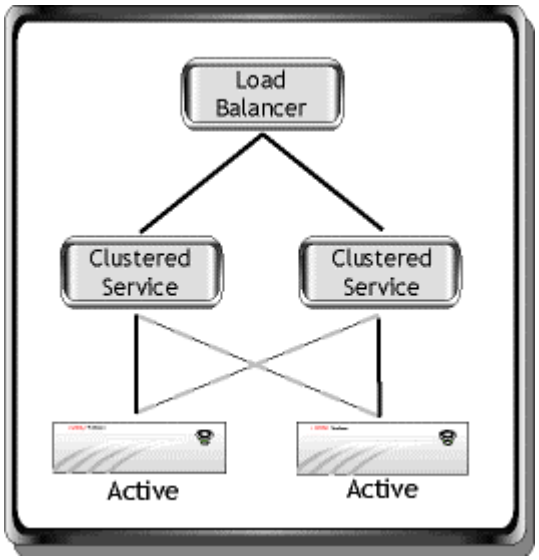

Figure 15: Load-balanced cluster

Not all servers in the figure are actually handling requests. Only the servers that have virtual proxy service addresses assigned to them are handling requests. For instance, if there are six TaskSmart C-Series servers and four virtual services, only four of the TaskSmart C-Series servers will be assigned any responsibility. The remaining two TaskSmart C-Series servers are awaiting a failure before they assume any responsibility.

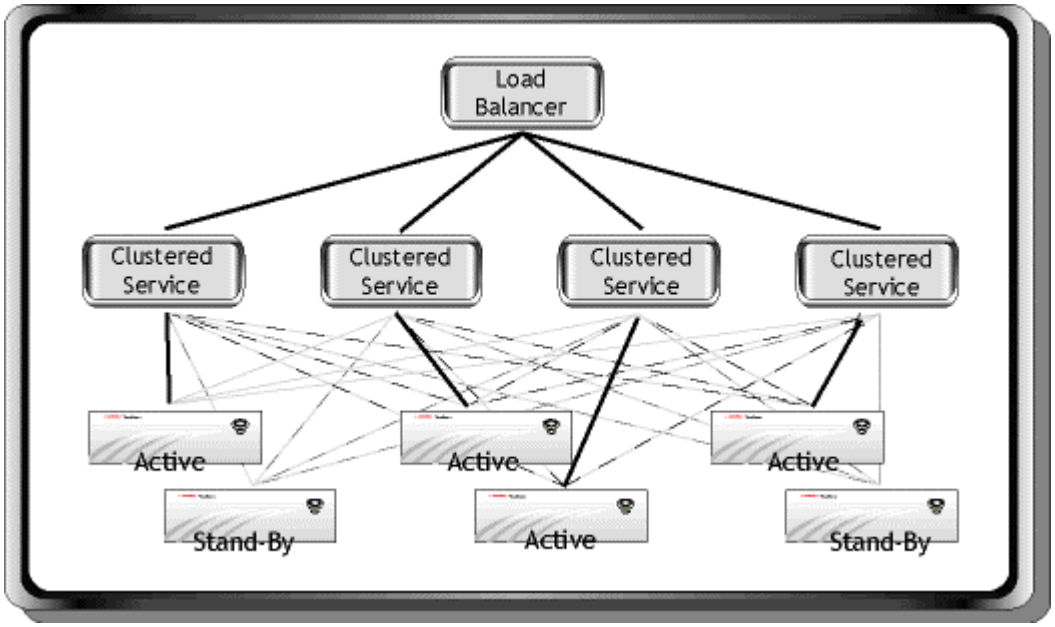

Figure 16: Load-balanced cluster with stand-by members

Security

A few areas draw security questions with the TaskSmart C-Series server. The most common question is in regard to secure client data. As mentioned, the TaskSmart C-Series server caching does not interpret this traffic. A properly deployed TaskSmart C-Series server will not interfere with or lessen the security of HTTPS traffic.

As far as security of the other objects in cache, only information that is publicly available on the Internet will be in cache. According to the HTTP specifications, after the TaskSmart C-Series server has established that the origin server is no longer reachable, objects can no longer be served from cache for that domain until the origin server begins responding.

The only information that should be considered sensitive is the actual configuration of the TaskSmart C-Series server. For this reason, TaskSmart C-Series provides for password protection on the Telnet configuration consoles. The address-based authentication allows administration only from certain client IP addresses or on specified TaskSmart C-Series server ports.

When deploying a TaskSmart C-Series server, use the practice of taking all available security precautions. Make sure to add passwords to the Telnet and FTP interfaces. Add authorized users' IP addresses to the authorized address list. Finally, restrict the addresses that can be used for configuration. This process is ICS based. For additional information, refer to *Compaq TaskSmart C-Series Servers Feature Procedures Guide*, document number 158D-0701A-WWEN.

Even with secure password logon to the configuration utilities and restricted IP lists for administration, Compaq does not recommend deploying the TaskSmart C-Series server outside of the firewall. Previously, this guide mentioned reasons for performance and proxy services, but there are security issues as well.

When deploying the TaskSmart C-Series server within a firewall with access restrictions, it may be necessary to have the TaskSmart C-Series server use a user name and/or a password to access through the firewall. TaskSmart C-Series servers support firewall logon via SOCKS v.4 (user name only) or SOCKS v.5 (user name and password). If the firewall does not support SOCKS, the firewall should be configured to restrict external access based on IP address.

If each client has a distinct SOCKS logon, the TaskSmart C-Series server cannot be placed between the clients and the firewall. It would be necessary to create a Demilitarized Zone (DMZ) where user SOCKS logons would be used to grant access to the TaskSmart C-Series server. This is analogous to the access control hierarchy with other proxies. The TaskSmart C-Series server cannot control access. Instead, the firewall or SOCKS compliant access control service must be able to communicate directly with the clients. After authenticating through a SOCKS device, the client request can then be sent to the TaskSmart C-Series server.

Conclusion

If a forward proxy better suits a particular network than transparent proxy does, this section should have answered most of the questions that determine the best location and configuration of the forward proxy. Some networks are too complex or unique to be considered in detail. Fortunately, the guidelines and considerations for deploying a TaskSmart C-Series server in simple networks also apply to networks that are more complex. If administrators and architects extend the guidelines and follow best practices, the TaskSmart C-Series server can be deployed as a forward proxy in any environment.

Transparent Proxy

This section of the guide provides all of the information that a system administrator or network architect needs to make an effective deployment of a TaskSmart C-Series server as a transparent proxy. Where relevant, the differences between the transparent deployment and the forward proxy deployment are highlighted.

This section addresses environments where the network is too large for manual configuration of the browsers and/or the network does not allow for client configuration. An ISP's dial-up client base is an example of a network that does not allow for client configuration. This example is where the only viable option is to deploy the TaskSmart C-Series server as a transparent proxy.

The term transparent is defined as it applies to the operation of the TaskSmart C-Series server. The *transparency* of the TaskSmart C-Series server applies only to the clients on the network. Even in transparent mode, the TaskSmart C-Series server is still functioning as a proxy. As a result, any devices that are involved in filling a client request will see the TaskSmart C-Series server as the source of the request. The TaskSmart C-Series server is actively intercepting and reissuing the requests. It is not a passive participant in filling the request. Any logging or analysis of a transparent proxy will still bear evidence of the presence of the TaskSmart C-Series server.

If the network is a point of publishing, with most traffic being outbound, skip this section and proceed to "Server Acceleration." If the network requires a forward proxy, read the previous section on forward proxy.

Introduction

Client acceleration as a forward proxy requires that the client browser actively direct all requests to the proxy. To have the browser redirect the request, the connection properties of the browser must be changed at each workstation. For a transparent proxy, however, there is no configuration necessary on the client since there are no browser settings. Instead, some other device is actively scanning network packets and redirecting any port 80-based HTTP traffic to the transparent proxy. There is no indication on the client workstation that the requests are being proxied.

Transparent proxy may be slightly less intuitive than forward proxy, because all applications of the proxy are handled through the network architecture rather than the browser configuration. Use a transparent proxy in an environment where configuring each of the clients is not possible.

There are some environments that a transparent proxy would not be a suitable solution. For example, a small office with few workstations where configuration could be managed manually would not be suitable. In this case, the additional cost of the redirecting hardware may not justify the additional performance afforded by the L4 switch, discussed in the following section. Note that with a transparent proxy, unlike a forward proxy, a client cannot change the browser configuration to attempt to circumvent the proxy.

In deploying a TaskSmart C-Series server as a transparent proxy, the TaskSmart C-Series server position and operation should maximize performance while delivering the required proxy services. For the caching service, there are two main goals:

- To improve network response to the clients
- To save bandwidth

Other goals may be associated with other proxy services. In many cases, other proxy services may play a role in sizing and placing a TaskSmart C-Series server. The other proxy services that relate to the TaskSmart C-Series server as a transparent proxy are filtering, logging, and access control.

Redirecting to the Transparent Proxy

Currently, there are several devices that can monitor, identify, intercept, and redirect an HTTP request. The most generic type of redirection device is an L4 switch. L4 refers to the "Layer 4" in the Open Systems Interconnection (OSI) model, also referred to as the transport layer. By examining the transport layer in a packet header, an L4 switch may determine the destination port and destination IP address for a particular packet. Based on a user-defined configuration, different types of traffic can be handled in a variety of ways. Web requests can be redirected to a TaskSmart C-Series server. FTP requests may be load balanced across a number of servers. A simple mail transfer protocol (SMTP) request can be redirected to a particular port on a particular firewall. There are many L4 switch manufacturers with numerous products from which to choose. Many offer an impressive feature list with many peripheral functions. All should be capable of performing the redirection necessary to enable the TaskSmart C-Series server to operate as a transparent proxy. Check with the switch manufacturer to confirm that a particular switch would be capable of performing the redirection necessary for a particular deployment.

In addition to the redirection, L4 and L7 (described in the following paragraph), switches offer other features that can make a more stable and more scalable proxy deployment. Most intelligent switches offer the ability to perform connection management to the redirection target. By limiting the number of connections to the redirection target, the likelihood of an overrun condition is greatly reduced. When configured to limit the number of redirected connections, the switch can then pass, deny, or delay additional connections.

An L4 switch is not the only option for performing the redirection to the transparent proxy. L7 switches are similar to L4 switches, but more powerful. They are capable of examining the application layer and seeing the requested URL. Based on the type of URL, an L7 switch can perform the same redirection on requests that should be forwarded to the TaskSmart C-Series server. It should suffice to know that an L7 switch is capable of redirecting requests that are cacheable for using the switch as part of a transparent proxy deployment. An L4 switch would redirect all port 80 traffic regardless of whether the traffic is cacheable.

Cisco routers are also capable of a form of redirection using the Cisco proprietary Web Cache Communication Protocol (WCCP, called Web Cache Control Protocol in version 1). Using WCCP, a Cisco router will redirect HTTP requests to one or more cache servers. L4 switching, L7 switching, and WCCP all perform the same function of intercepting and redirecting a client request to the TaskSmart C-Series server.

In each case the L4 switch, L7 switch, or WCCP-enabled router automatically redirects requests; therefore, there is no configuration necessary on the clients. This lack of client configuration is the primary difference between a forward and transparent proxy.

Transparent Proxy Operation

When deployed with a transparent proxy, clients will attempt to fill requests as though they are connected directly to the Internet. Clients will perform their own DNS resolution and submit requests through the default gateway to the origin server. A device intercepts the client request between the client and the origin server. The client request is delivered to the proxy after it is intercepted. This is done without requiring any interaction with the client. The proxy assumes responsibility for filling the request. If the request is held in cache, the proxy will respond. The client is unaware of the proxy. If the request was not held in cache, the proxy will fill the request on behalf of the client. The proxy will then forward the data to the client after creating a local copy.

Requests from cache will simply be filled more quickly to the clients. There may be a few changes noticeable to the end user other than the improved response. Clients might notice the **HTML alarm messages screen**, which is enabled by default. When the cache is operational, administrators can modify the messages sent to the users. These messages will be displayed only if the cache can not respond to the end user's request or another error occurs and the cache needs to alert the client.

Cacheable Data

Some types of HTTP traffic are not cached. Specifically, the TaskSmart server will not cache objects that are marked in the HTTP header as non-cacheable or carry implied non-cacheable.

Since the introduction of HTTP, there has been the ability to control intermediary caches from the origin server. Originally, this was done with the HTML header "PROGMA NO-CACHE." HTTP 1.1 extended the definitions of cacheable to include expirations and limited cacheable data. Additional information on cacheable objects can be found in the *Compaq TaskSmart C-Series Administration Guide*, Chapter 3.

Three types of objects that are not cached:

- Objects that are marked non-cacheable in the HTTP header
- Objects that exceed the maximum object size and are configurable variable on the TaskSmart C-Series server
- Objects that are implied as non-cacheable

Table 3. Cacheable Data

Type of Data	Is it cached?
HTTP	Yes, unless marked otherwise by origin server or is inherently non-cacheable.
HTTPS	No
Password-protected HTTP	Yes, but only served from cache to authenticated user.
Anonymous FTP	No, FTP is not cached by the Transparent Proxy.
Password-protected FTP	No
.ASP	No, requires server-side processing. Static embedded objects are cached.
.CGI	No, requires server-side processing. Static embedded objects are cached.
Cookies	No, origin server should mark cookies as “PRIVATE.”

A non-cacheable HTML object can be a document that is marked as non-cacheable or a document that implies that it is non-cacheable. Implied non-cacheable is any HTML that requires server-side processing, such as .CGI or .ASP. This does not mean that the TaskSmart C-Series server will not accelerate .CGI or .ASP pages. TaskSmart C-Series server caching is done at the object level, not the page level. This means that all static images (.JPG, .BMP, .GIF, etc.) are cached. In a given .ASP HTML document, there may be many static cacheable objects. The static objects are the same every time the page is viewed, such as a corporate logo. These objects will be cached and they will be served from the high-speed local cache when they are next requested as part of a dynamically generated HTML document. In terms of bandwidth, the embedded objects and images that are static and cacheable represent a considerable and large percentage of the total size of a Web page. Even in dynamic pages, there is a large portion of data that is cacheable.

Another type of data that is not cached is hypertext transport protocol secure (HTTPS) or secure data. In fact, HTTPS data is simply passed through the TaskSmart C-Series server. Any banking transactions or online purchases are essentially transparent to the TaskSmart C-Series server. Because HTTPS uses port 443, the proxy services are not applied to secure sessions. Secure data should not be confused with private or password-protected data. Data that is static and requires a user name and password to obtain is cached. If a user name and password are required, the objects are marked so that only the authenticated user can view them from the cache. Therefore, they are cached only for the single user. If another user uses a different logon to view the same information, the information will not be served from cache. The user name and password are not cached and the authentication is still performed on the origin server. The origin server generally marks cookies as non-cacheable or “private.” Cookies should always be marked private if they contain user-specific data. Any objects that are marked “private” cannot be held in any cache other than the client browser.

Some objects may be cached, but for only short periods of time. In these cases, an origin server may specify an expiration time for an object in the HTTP header. After the expiration time has passed, the TaskSmart C-Series server must retrieve the object from the origin server and discard the one held in cache when the object is next requested. The TaskSmart C-Series server is not allowed (by the HTTP specifications) to vend out objects that it believes may be expired. This means that if an object has expired and the origin server is not available, the TaskSmart server will not display the expired data.

Architecture and Placement in a Network

Undoubtedly, the most common question concerning deployment of a TaskSmart C-Series server is "Where do I put it in my network?" There are three basic requirements for the most effective placement of a TaskSmart C-Series server in a particular network.

- The TaskSmart server should be placed logically between the clients and the Internet. Clients will submit requests to the TaskSmart C-Series server and the server will respond with a response it has received from the Internet. The most logical and efficient placement would then be between the client and the Internet. Placing the TaskSmart C-Series server between the client and the Internet also reduces redundant traffic on segments that connect to the Internet.
- Place the TaskSmart server as close to the clients as possible. Requests should be sent *upstream* to the TaskSmart server. The number of hops from the workstation to the Internet should be minimized. This architecture reduces latency and eliminates potential points of failure. After this practice is established and understood, the architect should then try to reduce the number of hops between the client and the TaskSmart C-Series server. As the cache hit ratio increases, more objects will be served from cache. There will be a greater volume of traffic between the end users and the TaskSmart C-Series server than the TaskSmart C-Series server and the origin servers. Essentially, this minimizes potential bottlenecks between the TaskSmart C-Series server and the Internet.
- Place the TaskSmart server as far from the clients as performance and savings allow. In some cases, the greatest return on the investment for a TaskSmart C-Series server will mandate that several hops lie between the TaskSmart C-Series server and a client base that is large enough to generate a substantial amount of Web requests.

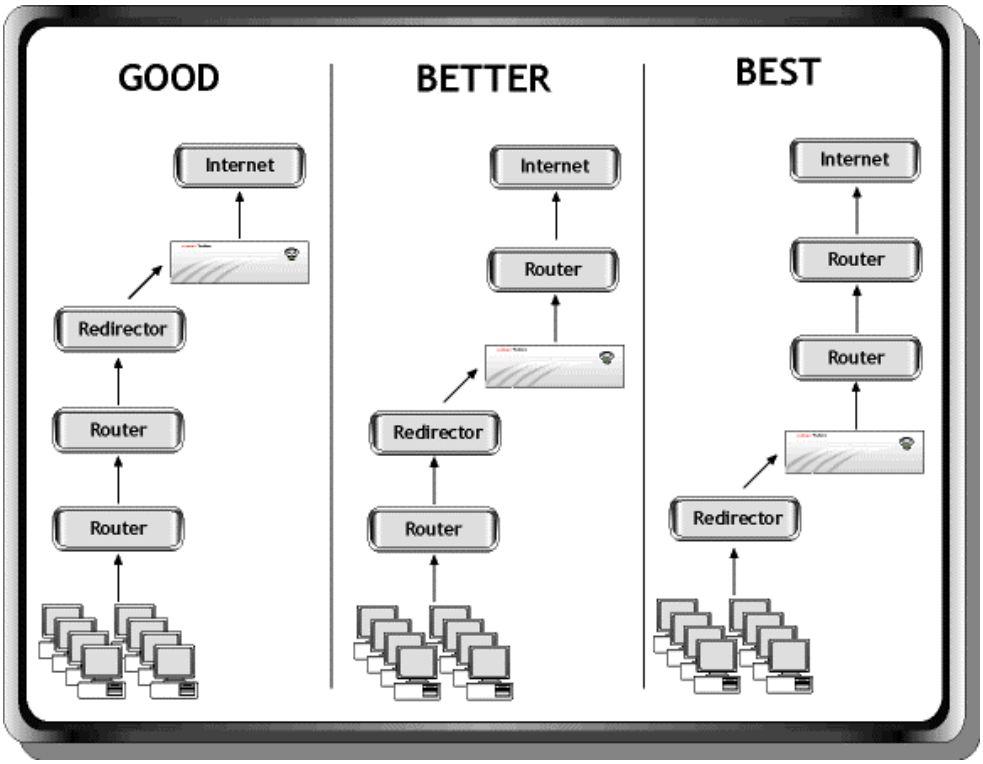

Figure 17: Minimize hops and place close to clients

Generally, the locations in the network that meet these requirements are points where users are funneled together (*aggregation point*) within the network or transferred to another network (*access point*). Placing the TaskSmart C-Series server at a location where users are funneled allows the deployment to best match the TaskSmart C-Series server's rated load. Because there may be more than one of these *aggregation points* in a network, the TaskSmart C-Series server should be deployed at the point that best matches its rated load. Deploying the TaskSmart C-Series server at an *access point* maximizes the number of potential users of a single larger cache.

Keep in mind that when this guide discusses placement of the TaskSmart C-Series server, this may or may not also include the redirecting device. In the case of an L4 or L7 switch, the TaskSmart C-Series server must be connected directly to a port on the switch. For the L4 and L7 switches, the redirection is not IP-based and therefore cannot be routed between the redirector and the TaskSmart C-Series server. For a transparent proxy using Cisco WCCP to perform the redirection, the TaskSmart C-Series server does not have to be placed logically or physically adjacent to the Cisco router. Additional hops between the redirecting WCCP router and the TaskSmart C-Series server will add latency to every request.

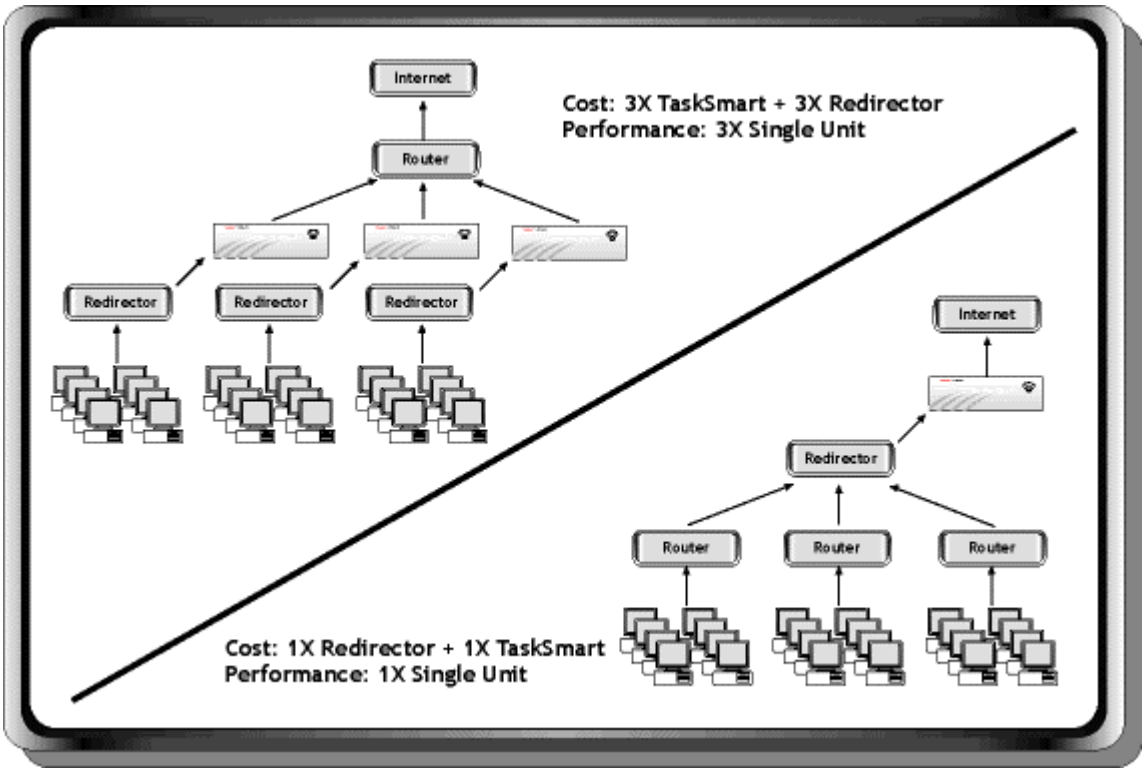

Figure 18: Matching loads to placement

Physical Connections

Questions may remain concerning implementation of the architecture through physical connections. Obviously, a requirement of the physical connections is that they must provide stable connections to all necessary resources. The TaskSmart C-Series server must be able to reach the default gateway for proper operation. All other network resources must be accessible directly or through a network gateway on either one or more of the physical connections. These other resources include clients, servers, and DNS services.

It is possible to use multiple interfaces in your configuration. Using the primary interface for internal connection and the second interface for public access is a common approach. Non-routable network addresses can increase security for internal networks and servers. This also allows the TaskSmart C-Series server to retrieve data from origin servers. The physical interfaces should provide the maximum bandwidth necessary for your configuration. A gigabit Ethernet NIC option is available if it is required. Refer to the *Compaq TaskSmart C-Series Servers Performance Guide* # 155E-0701A-WWEN to review your sizing requirements and capabilities

Transparent Proxy TaskSmart Hierarchies

In some transparent deployments, the proxy service requirements may dictate that the TaskSmart C-Series server should be deployed along with another TaskSmart C-Series server in a hierarchy. A TaskSmart C-Series server deployed as a transparent proxy can provide the same proxy services that are available from a forward proxy deployment for logging, filtering, and caching. Readers for which a single TaskSmart C-Series server alone can provide all of the required performance, logging, filtering, and caching services should skip this section.

This topic assumes that the network has proxy requirements that demand multiple levels of caching and that TaskSmart C-Series servers can provide all necessary proxy services. If the TaskSmart C-Series server must leverage the proxy services of a non-TaskSmart C-Series server proxy, then skip to "Transparent Multi-Proxy Hierarchies." If the network is small with only one proxy necessary to meet all performance and proxy service requirements, skip to "DNS Access, Firewalls, and Routers."

Because the architecture, operation, and deployment of a transparent proxy in a hierarchy is identical to that of a forward proxy, there will be very little difference between the forward proxy hierarchy discussion and that of transparent proxy hierarchies.

Hierarchy Relationships

Alone, a single-level TaskSmart C-Series deployment reduces redundant traffic generated by users. As a hierarchy, groups of TaskSmart C-Series servers can not only reduce redundant traffic for users, but also reduce the redundant traffic among *groups* of users, with each group connected to its own proxy. The bandwidth savings of a TaskSmart C-Series hierarchy can be substantially higher than a single-level deployment in the same environment.

In a hierarchy, TaskSmart C-Series servers that are nearer to the clients are called *children*. The servers that are one level up in the hierarchy are *parents*. In a three-level cache, there are no grandparents. Instead, the parents of the first hierarchy are now considered the *children* in a new hierarchy.

While a network may have multiple levels, there are only two logical levels in any hierarchy-parent and child. Each hierarchy is considered independent of levels above or below it. This means that in a hierarchy, even if there are four levels, only two of them are listed in any single hierarchy configuration.

In hierarchies consisting exclusively of TaskSmart C-Series servers, there are only three possible relationships:

- ICP Parent
- ICP Child
- ICP Peer

ICP Parent

When deployed as an ICP parent, the TaskSmart C-Series server will respond to requests that are submitted from proxies that are closer to the clients. An ICP parent does not need to know the addresses of the child caches below it. The children must be aware of the parent's address and the parent must know only that it is a parent. As an ICP parent, a TaskSmart C-Series server would have one or more caches from which it receives requests when the child caches could not be filled. In some cases, the ICP parent will be required to fill the request on behalf of the client. In this situation, the requested object would then be held in both the child and parent caches for future use.

ICP Child

As an ICP child, the TaskSmart C-Series server will forward all requests to an upstream parent cache before attempting to fill the request. An ICP child can also be configured to fill all requests through the parent, even if the parent does not already have the object in cache. The configuration necessary for an ICP child is simply specifying an ICP parent. The ICP child is not an explicit hierarchy entity.

ICP Peer

ICP-peered caches are similar to the relationship between children and parents, except that peers cannot be configured to fill ICP misses through their peers. If an ICP peer requests an object of a peer and receives an ICP "miss" reply, the requesting peer must either fill the request itself or forward the request to any specified parents. ICP peering is not necessarily bi-directional. TaskSmart C-Series server A may have TaskSmart C-Series server B as a peer, but TaskSmart C-Series server B does not need to have TaskSmart C-Series server A as a peer.

Any level in a hierarchy may have peers. TaskSmart C-Series servers can be children of the same parent and thus peers to each other. When a request reaches a TaskSmart C-Series server that has both a peer and a parent, the TaskSmart C-Series server will always check to see if the peer has the requested object before forwarding the request to a parent. Peering of TaskSmart C-Series servers is not automatic. Just because two TaskSmart C-Series servers are children of the same parent, it does not mean they operate as a peered hierarchy. Peers must be listed in the hierarchy configuration just as with parents.

TaskSmart C-Series servers issue requests to peers and parents via the Internet Cache Protocol (ICP). ICP is a widely accepted protocol and is supported by nearly all caching proxies. This means that cache hierarchies can be built with other brands of caching solutions. For more information on the TaskSmart C-Series server deployment with other proxies, see "Forward Multi-Proxy Hierarchies." For this topic, hierarchies will consist only of TaskSmart C-Series servers.

Architecture and Placement in a Hierarchy

Unlike the concept, the configuration for a hierarchy is very simple. Simply indicating the addresses of the parents and any peers in the hierarchy are all that is necessary on the TaskSmart C-Series server. Positioning parents in relationship to children can be considered the same way placement is decided with a single TaskSmart C-Series server.

- The parent server should be placed logically between the children and the Internet. Any requests that the child cache attempts to fill from a parent should travel towards the Internet and the origin server. Placing a parent cache farther within the network will create additional internal traffic and delays to filling the client request.

- Place the parent server as close to the children as possible. Reducing hops and distance between members of a hierarchy will reduce latency between them. Keep in mind that the concept and performance of a hierarchy implied some manner of a large distributed network. As a result, there may be several hops between siblings. Try to reduce unnecessary hops.
- Place the parent server as far from the children as performance and savings allow. To better match loads on the TaskSmart C-Series servers and thereby maximize the return on investment, it may be necessary to add one or more hops between the parent cache and the child cache. This placement will result in the TaskSmart C-Series server getting additional requests.

TaskSmart C-Series Proxy Services in a Hierarchy

This section details deployment options and considerations for networks that will use only TaskSmart C-Series servers to provide the proxy services in the hierarchy. Because all TaskSmart C-Series servers can provide the same proxy services, the relative placement of the services will be determined more as a matter of performance or convenience.

Authentication

The TaskSmart C-Series servers provide LDAP user authentication if required. All forms of plain text authentication will work with a transparent TaskSmart C-Series server hierarchy. HTTPS operating through secure sockets layer (SSL) can be proxied by the TaskSmart C-Series server. If Windows NT Challenge/Response (NTLM) authentication is used on the Web server, then no member of the hierarchy should proxy the client request to that server. Instead, the browser should be configured to bypass the proxy for requests bound for an NTLM authenticating server.

Access

The TaskSmart C-Series servers support IP address and port-based restriction. The TaskSmart C-Series server can restrict services based on IP address. If IP-based access control is used, be sure that no Network Address Translation (NAT) equipment can obfuscate the end user's IP address.

Logging

Logging requires that the log service be deployed at the level that allows the proper resolution. Applying the logging service as a parent allows logging of the child proxy requests only. For logging at the user level, the logging service must be applied at the proxy closest to the clients. Logging at the parent level in a hierarchy would show only the children issuing requests, unless another group of users is accessing the parent directly.

Filtering

The initial release of the TaskSmart C-Series servers will not provide content filtering services. These features should be supported in a later release of the product. Proxy filtering services block users from viewing material that has been deemed inappropriate. Filtering services can differ in how they determine which content should be blocked. The most common methods of determining inappropriate content are a user-defined list of URIs, a rating system, and a subscription to a filtering service that maintains lists of URI classifications.

In some cases, the filtering requirements can dictate the optimal location and configuration of a client accelerator. Filtering can be deployed at different points in the hierarchy to enable either user-level or group-level filtering. If all users are subject to the same filters, it does not matter which level of the hierarchy is handling the filtering service. If all requests are forced through the hierarchy, the filtering service is being applied to all users.

Practicality may force one level or another to handle the filtering service because of filtering methods. The only compelling reason for placing filtering at one level or another is that filtering at the child level allows for different filter sets to be defined for each child, but requires that two filter lists be administered. Filtering at the parent level requires only one filter list, but all clients that access the Internet through the parent will be subject to the same filter set. That includes both through the children and directly through the parent. Filtering can be done at both the parent and child level, allowing distinct group filter sets on each child and a parent filter list that applies to all child user groups.

Caching

Caching is different from the other proxy services. Caching is the only one that can benefit by having multiple layers of redundant service. Therefore, if TaskSmart C-Series servers are deployed in a hierarchy, use all available services to create a caching hierarchy that will increase the bandwidth savings.

Sizing a Hierarchy

A hierarchy should be sized such that the parent should be able to handle all requests that cannot be filled by all children. Since each hierarchy is only aware of two levels, there is no practical limit on the number of levels to the architecture. A four-level architecture of TaskSmart C-Series servers is more than capable of handling the requests for tens of thousands of users. Therefore, it is unlikely that a network would ever have more than four levels in the proxy hierarchy architecture. After the hierarchy is configured, the cache monitoring tabs on the graphical user interface can provide details on the number of requests that were received through the hierarchy using ICP or HTTP parent caching.

If there is any traffic that does not originate from the children, then this must be taken into account when sizing the TaskSmart C-Series server. A child proxy can also have more than one parent proxy. All parents are queried for objects with more than one parent. Whichever parent responds first will fill the child server request if the object is in more than one parent cache. The second response is ignored.

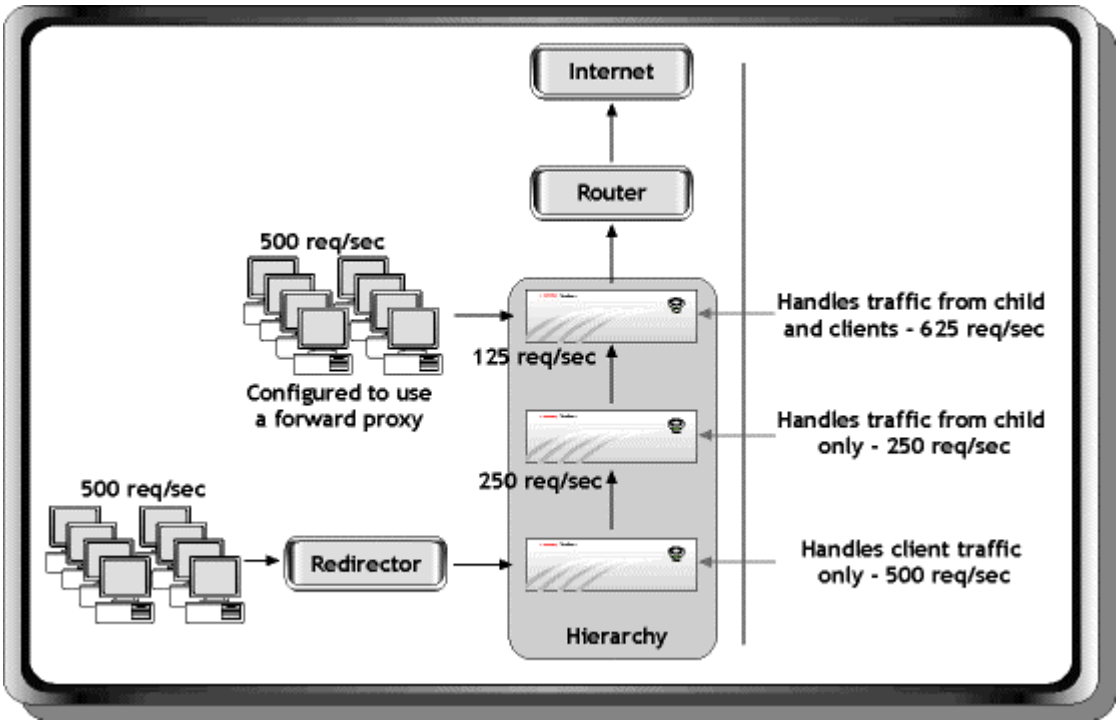

Figure 19: Sizing the hierarchy

Transparent Multi-Proxy Hierarchies

This topic attempts to provide all of the information needed to make an effective deployment of a transparent proxy hierarchy consisting of both TaskSmart C-Series servers and non-TaskSmart C-Series server proxies. The methods and architecture for deploying TaskSmart C-Series servers as part of a hierarchy with other proxy servers are explained. It is assumed that the network has proxy service requirements that the TaskSmart C-Series server cannot meet alone and that TaskSmart C-Series servers will not be used at every level in the hierarchy. If the network is small with only one proxy necessary to meet all performance and proxy service requirements, skip to the "DNS Access, Firewalls, and Routers."

Many proxy hierarchies will have to rely on a non-TaskSmart C-Series server proxy to provide a necessary proxy service. In this case, care should be taken to select the proper relationships, placement, and protocols used in the hierarchy. There are several methods for a successful deployment. One architecture is no better than another architecture. The deployment that has the greatest return for the environment is the correct way to deploy the TaskSmart C-Series server. The return can be measured by performance, cost, or a combination of both. It is important to understand the goals and expected return. Perhaps the first aspect to understand is the services that the current proxy is providing. When there is a clear understanding of the current proxy's role, the optimal location for the TaskSmart C-Series server is easier to determine.

Heterogeneous Hierarchical Relationships

When building a hierarchy, proper operation and optimal performance will rely on the correct protocols used to forward requests between the members of the hierarchy. Just as with TaskSmart C-Series server hierarchies, a proxy hierarchy consists of children and parents. Logically, children are closer to the clients than parents. Operationally, children submit requests to parents but parents do not submit requests to the children.

Every type of hierarchy must have a method of forwarding requests. The TaskSmart C-Series can use ICP to forward requests to parents. A TaskSmart C-Series server can receive requests on the standard proxy (port 8080) interface, the standard HTTP request (port 80) interface, from an ICP child. Port 80, when required, must be enabled in the TaskSmart C-Series server. Since ICP is an open standard supported by nearly all proxy caches, the ICP protocol will provide the greatest flexibility for future growth when building a hierarchy. The most notable exception to the list of proxies that does not support ICP is Microsoft Proxy 2.0.

There are only five possible hierarchical configurations for TaskSmart C-Series servers:

- ICP Parent
- ICP Child
- ICP Peer/Sibling
- HTTP Cache Hierarchy

ICP Parent

When deployed as an ICP parent, the TaskSmart C-Series server will respond to requests that are submitted from proxies that are closer to the clients. An ICP parent does not need to know the addresses of the child caches below it. The children must be aware of the parent's address and the parent must know only that it is a parent. As an ICP parent, a TaskSmart C-Series server would have one or more caches from which it receives requests when the child caches could not be filled. In some cases, the ICP parent will be required to fill the request on behalf of the client. In this situation, the requested object would then be held in both the child and parent caches for future use.

ICP Child

As an ICP child, the TaskSmart C-Series server will forward all requests to an upstream parent cache before attempting to fill the request. An ICP child can also be configured to fill all requests through the parent, even if the parent does not already have the object in cache. The configuration necessary for an ICP child is simply specifying an ICP parent. The ICP child is not an explicit hierarchy entity.

ICP Peer

ICP-peer caches are similar to the relationship between children and parents, except that peers cannot be configured to fill ICP misses through their peers. If an ICP peer requests an object of a peer and receives an ICP "miss" reply, the requesting peer must either fill the request itself or forward the request to any specified parents. ICP peering is not necessarily bi-directional. TaskSmart C-Series server A may have TaskSmart C-Series server B as a peer, but TaskSmart C-Series server B does not need to have TaskSmart C-Series server A as a peer.

HTTP Cache Hierarchy

When deployed as a "parent" to a Microsoft Proxy server, the Microsoft Proxy server must have the TaskSmart C-Series server listed as an HTTP Cache Hierarchy. The TaskSmart C-Series server is in other words an "upstream Web proxy." This configuration is possible only on Microsoft Proxy. When configured to use an HTTP Cache Hierarchy, Microsoft Proxy will submit requests that it cannot fill locally to the TaskSmart C-Series server. To the HTTP Cache Hierarchy, these requests are formatted just as a proxied client request. For a TaskSmart C-Series server to function as an HTTP Cache parent, the forward proxy service must be enabled.

Protocols in a Multi-Proxy Hierarchy

ICP Hierarchies

For an ICP hierarchy, child servers need know only the ICP address and port for the parent servers. Squid, Cisco, BorderManager, Infolibria, CacheFlow and many others support ICP. For configuring each of these servers to use ICP within a hierarchy, refer to the appropriate server documentation.

Architecturally and operationally, an ICP hierarchy of different types of proxy caches is identical to an ICP hierarchy of TaskSmart C-Series servers. The ICP protocol is a standard that should not differ from one proxy to another.

HTTP Parent Cache Hierarchies

TaskSmart C-Series servers often need to be deployed along with a non-ICP compliant proxy. The most notable configuration is the Microsoft Proxy and TaskSmart C-Series server hierarchy. Based on the discussion of the proxy services, it should be clear whether the TaskSmart C-Series server should be between the clients and the Microsoft Proxy server or between the Microsoft Proxy server and the Internet. Unlike ICP hierarchies, the configurations for the different placements are quite different.

If the TaskSmart C-Series server is between the Microsoft Proxy server and the Internet, then the TaskSmart C-Series server should be configured as a standard forward proxy with no special hierarchical configuration. Microsoft Proxy requires either configuration to indicate an upstream Web proxy, or no configuration and a redirector. If the configuration indicates an upstream Web proxy, then neither the TaskSmart C-Series server nor the Microsoft Proxy servers are transparent. If the redirector is used then the TaskSmart C-Series server is transparent and client configuration must still point to the Microsoft Proxy server to forward requests to another proxy. The Microsoft Proxy server may still control user access to the Internet and all authorized users will pass through the TaskSmart C-Series server where the proxy services provided by the TaskSmart C-Series server can be applied to authorized users.

The TaskSmart C-Series server can see only the Microsoft Proxy server and not the clients in this configuration. Therefore, any logging of client IP addresses would have to be done by the Microsoft Proxy server. Another item to note in this configuration is that the performance of the TaskSmart C-Series server will be limited to that of the Microsoft Proxy server.

For the other scenarios where user-level resolution is not necessary, reversing the logical positions of the Microsoft Proxy server and the TaskSmart C-Series server can be done. The Microsoft Proxy server may still provide plain text-based user-level access control, filtering, caching, and logging.

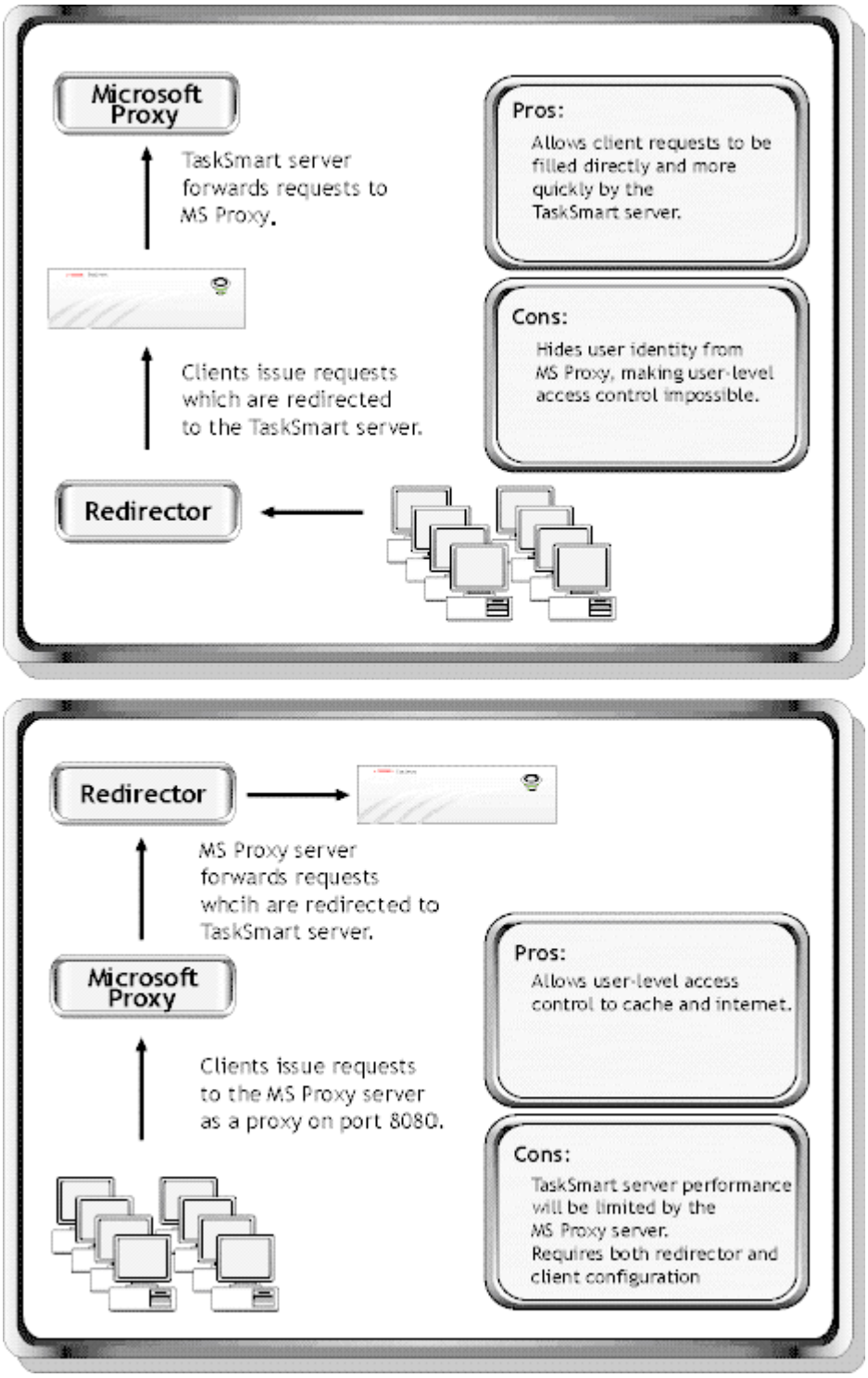

Figure 20: Transparent deployment with a Microsoft proxy server

With the TaskSmart C-Series server placed between the clients and the Microsoft Proxy server, the Microsoft Proxy server requires no extra configuration beyond a standard forward proxy configuration. On the TaskSmart C-Series server, however, we must set up a hierarchy analogous to the upstream Web proxy required in the previous configuration. In order to forward the requests to the Microsoft Proxy server, configure the TaskSmart C-Series server to consider the Microsoft Proxy server as an HTTP parent.

Architecture and Placement in a Multi-Proxy Transparent Hierarchy

This section details deployment options and considerations for networks that will use servers other than TaskSmart C-Series servers. There are two variables in a proxy hierarchy deployment:

- Distribution of proxy services between the proxies
- Relative placement of the two servers

Generally, the service for the additional proxy that is needed will play a key role in determining both the distribution of services and the relative placement of the proxies.

Authentication and Access Control

When building hierarchies of different types of proxies, the architecture cannot be reduced to simple guidelines. In most cases, the reason for the mixed hierarchy will be to deliver a proxy service that one proxy does not offer. The relative placement of the proxies may enhance or inhibit the delivery of proxy services for access control, filtering, and logging. In this section, each proxy service and the implications of the relative placements will be discussed.

Common Deployment for all Proxy Hierarchy

Place the TaskSmart C-Series server as close to the clients as possible. Just as before, placing the TaskSmart C-Series server closer to the clients eliminates latency on the response to the client, reduces propagation of redundant traffic through the network, and eliminates potential bottlenecks between the TaskSmart C-Series server and the clients. Also place the TaskSmart C-Series server as far from the clients as performance and services require. Some proxy service requirements will dictate the most effective location for the TaskSmart C-Series server.

Hierarchy Architecture with an Access Control Proxy

The TaskSmart C-Series server should be as close to the clients as performance and services allow. The easiest placement to consider is dictated by access control. Therefore, the distribution of access control services is decided rather easily, the TaskSmart C-Series server can provide access control services for both LDAP and IP access control. The proxy providing the access control service must be connected directly to the clients.

When using the Windows NT Challenge/Response (NTLM) authentication, the TaskSmart C-Series server cannot sit between the client and the authenticating server. NTLM does not support authentication through a proxy. Therefore, if NTLM will be used, the Microsoft Proxy server must be placed closer to the clients than the TaskSmart C-Series server. No one configuration is better than the other. The correct configuration is the one that allows the proper delivery of proxy services.

Hierarchy Architecture with a Logging or Filtering Proxy

Logging also requires that the log service be deployed at the level that allows the proper resolution. Applying the logging service as a parent allows only logging of the child proxy requests. A parent in a hierarchy cannot distinguish between clients on the other side of its own children. For logging at the user level, the logging service must be applied at the proxy closest to the clients.

Likewise, filtering can be deployed at different points in the network to enable either user-level or group-level filtering. If all users are subject to the same filters, it does not matter which level of the hierarchy is handling the filtering service. The filtering service is being applied to all users if all requests are forced through the hierarchy.

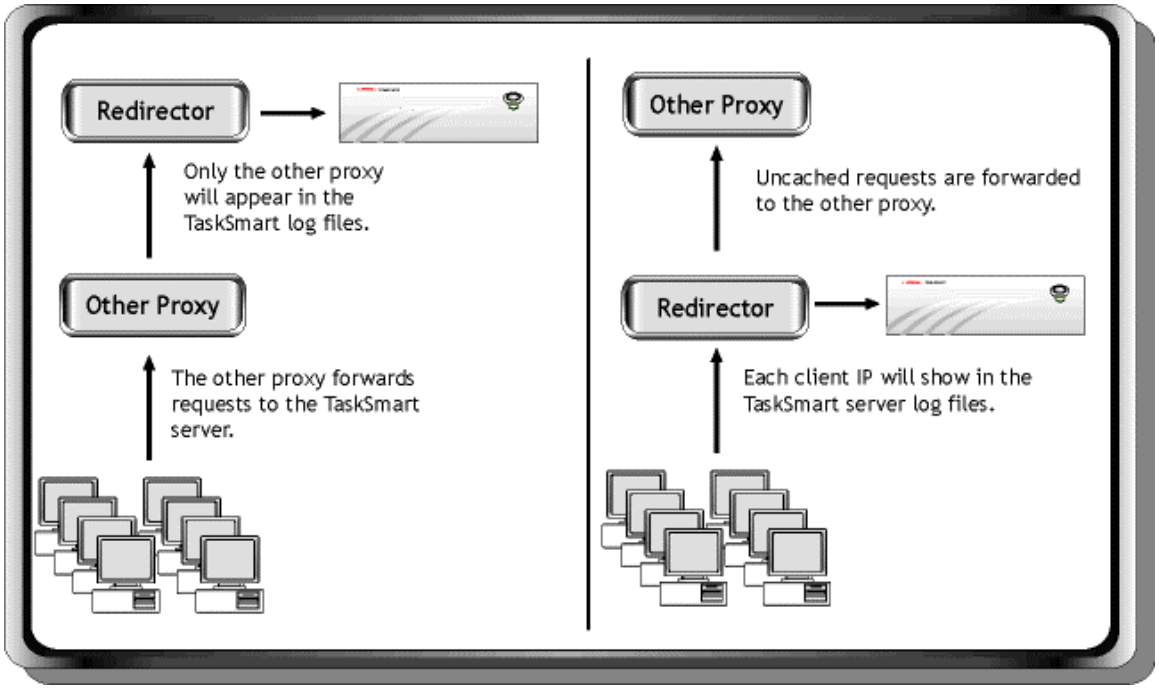

Figure 21: Effect of logging on placement

Practicality may force one level or another to handle the filtering service because of filtering methods. Proxy filtering services block users from viewing material that has been deemed inappropriate. Filtering services can differ in how they determine which content should be blocked. The most common methods of determining inappropriate content are a user-defined list of URIs, a rating system, and a subscription to a filtering service that maintains lists of URI classifications.

For all proxy services to work properly in a forward proxy environment, the firewall must be configured to not accept requests directly from the clients. It is simple for a client to circumvent all proxy services simply by changing their browser configuration if the firewall is not blocking client access. Following these guidelines will create an architecture that makes the most effective use of the TaskSmart C-Series server high-speed cache and will meet all of the filtering requirements.

Hierarchy Architecture with a Caching Proxy

Caching is different from the other proxy services. Caching is the only one that can benefit by having multiple layers of redundant service. Use both services to create a caching hierarchy similar to the caching hierarchies of TaskSmart C-Series servers if the TaskSmart C-Series server is deployed in a hierarchy with another proxy that has its own caching service.

Performance rather than proxy service requirements will dictate the relative positioning of the TaskSmart C-Series server and another server providing caching services. Because the TaskSmart C-Series server is a single-function device that has been tuned for the proxy caching service, it will outperform most other proxy packages. There is the potential for the other proxy to become a bottleneck to the higher speed TaskSmart C-Series server caching service if another proxy is deployed between the clients and the TaskSmart C-Series server. Performance would be greatest if the TaskSmart C-Series server were closer to the clients than the other proxy.

Sizing a Multi-Proxy Hierarchy

A multi-proxy hierarchy should be built and sized just like a single-server deployment. Make sure that each TaskSmart C-Series server or other proxy can effectively handle all incoming requests from both the clients on the network and any children in the hierarchy. If there is any traffic that does not originate from the children, then this must be taken into account when sizing the TaskSmart C-Series server.

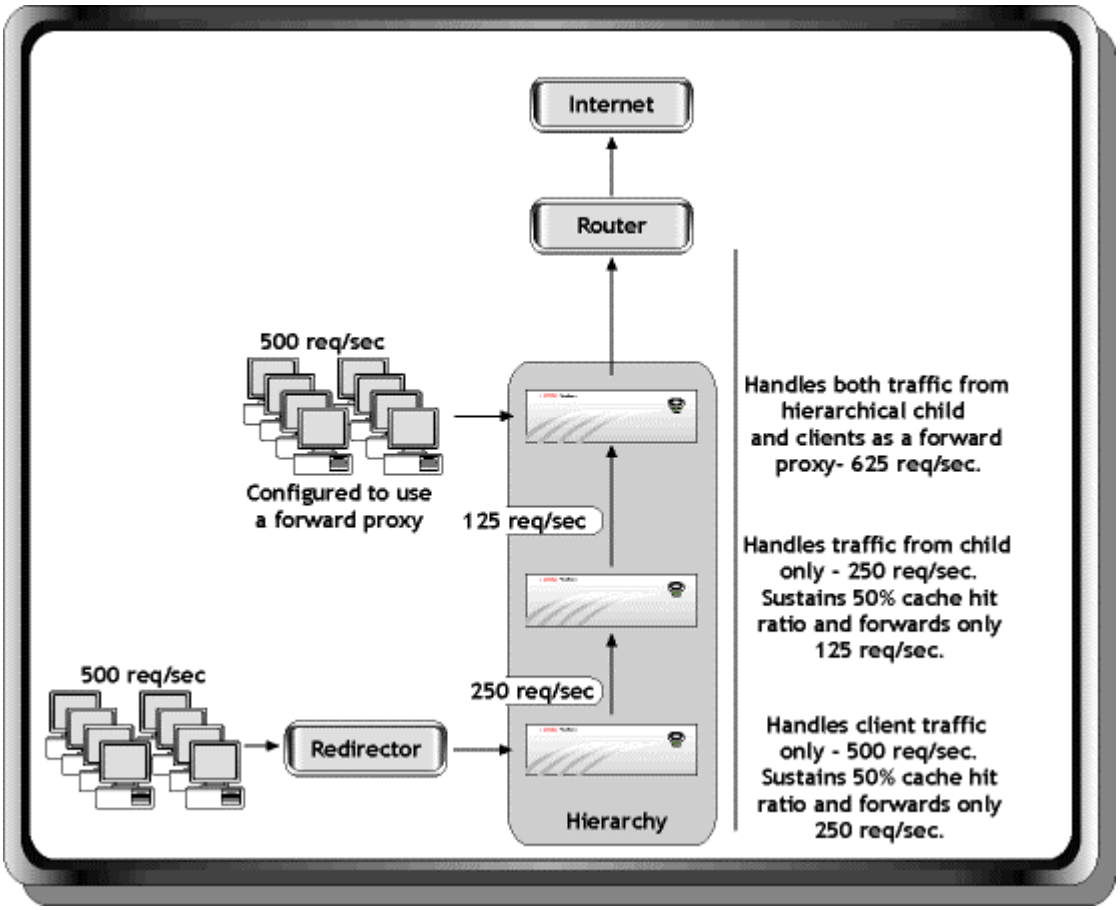

Figure 22: Sizing the hierarchy

To maximize the request rate to the TaskSmart C-Series server, administrators will need an understanding of traffic patterns at aggregation points in the network. Ideally, the TaskSmart C-Series servers should be deployed at the largest available aggregation points. This would provide the greatest probability of request overlap. For example, if a network were composed of many small user groups with each having its own Squid proxy, the point of the largest request rate would be between the Squid proxies and the Internet. The placement of the TaskSmart C-Series server should be as a parent to the Squid proxies. If the same client base shares a single proxy, then the TaskSmart C-Series server should sit between the clients and the Squid proxy, as a child to the Squid proxy. In both cases, the position would place the TaskSmart C-Series server as close to the clients while maximizing the incoming request rate to the TaskSmart C-Series server.

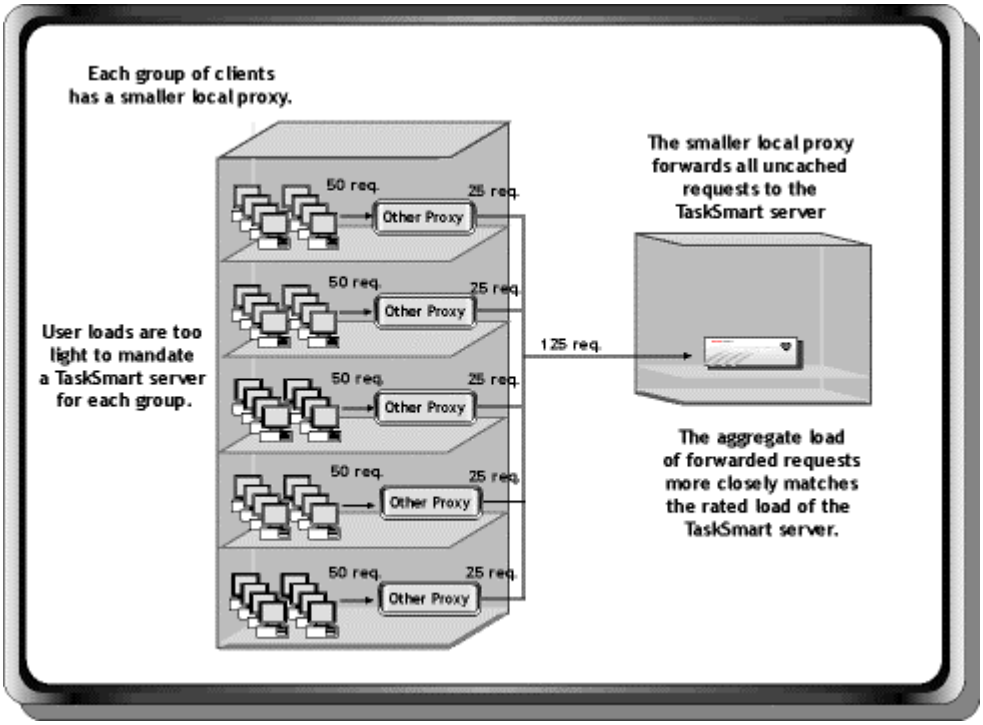

Figure 23: Mixed hierarchy with sizing considerations

A common configuration issue with hierarchies arises when the firewall allows only the parent to perform external DNS. By default, the TaskSmart C-Series server will perform its own DNS resolution and forward the requests with DNS already resolved. The TaskSmart C-Series server will not be able to resolve any DNS lookups that require external DNS if the firewall is set to allow only the parent server to access the external network. To force the TaskSmart C-Series server to send its requests through the parent proxy, enable the **CONFIG proxy.config.http.no_dns_just_forward_to_parent 1** option in the **records.config** file. The TaskSmart C-Series server manual covers this in depth. In doing so, the TaskSmart C-Series server will fill all requests through its parents in the hierarchy. Be sure to also set the parent proxy in the **Traffic Server** configuration utility administrator. Enter the name or IP address followed by a colon and the port number of the parent proxy.

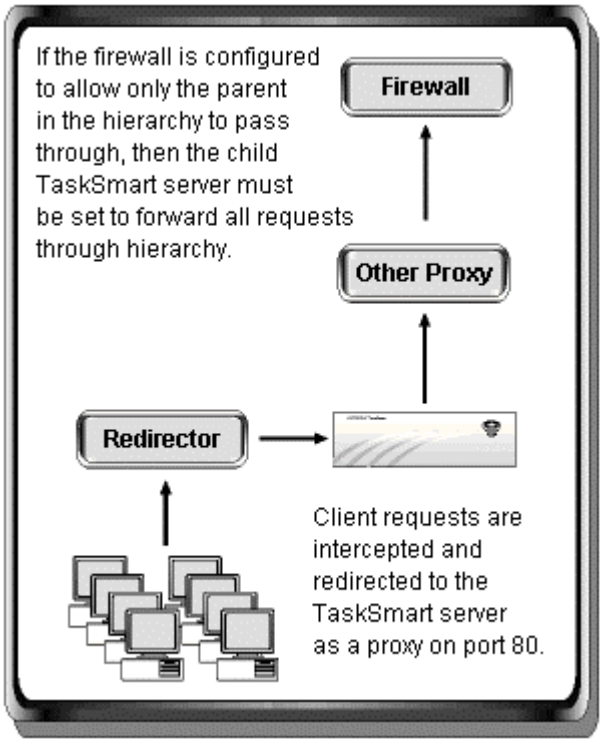

Figure 24: "Must only forward through hierarchy" scenario

Generally, there is no reason for caches in the network to have to climb the hierarchy for Intranet access or local content. If, however, the TaskSmart server is forced to fill only through the hierarchy DNS requests for local machines will fail because the DNS requests are being forced to the external DNS service. To correct this, TaskSmart servers allow the user to enter domains, addresses, or URLs that can be accessed directly by bypassing the proxy hierarchy.

The Microsoft Proxy server configuration is unique, as Microsoft Proxy 2.0 does not support the ICP protocol. A host of other proxies and caches support ICP, providing the ability to architect overlap within the caching framework. Making efficient use of hierarchies can drastically increase bandwidth savings in larger networks.

DNS Access, Firewalls, and Routers

For all proxy services to work properly in a transparent proxy environment, clients must be able to perform DNS resolution for all valid requests. This may require modifications to both the firewall or DNS servers. In some cases, the firewall and DNS servers may already allow both the TaskSmart C-Series server and clients to perform all DNS lookups.

Use the firewall not only to limit internal access, but also to prevent unauthorized users from taking advantage of an exposed cache. While the cached data inside of the TaskSmart C-Series server may hold little or no inherent value, the proxy services of the TaskSmart C-Series server itself should be protected from unwanted external traffic. Any users that are able to ping (packet Internet groper) the TaskSmart C-Series server could potentially use the proxy services. In extreme cases, random and unauthorized users on the Internet could use the high-speed caching service of the TaskSmart C-Series server to accelerate their access while consuming bandwidth and resources meant for your clients. For this reason, the TaskSmart C-Series server should generally be deployed within a firewall. At the same time, this will generally place the TaskSmart C-Series server closer to the clients.

For most transparent proxy TaskSmart C-Series server deployments, there should be no other configuration necessary on any routers, switches, or hubs. Following best practices of minimizing latency and reducing hops between clients and the Internet is all that is necessary to make the most effective deployment of the TaskSmart C-Series server.

Performance of a Transparent Proxy

This topic provides all of the performance information and considerations that could impact a TaskSmart C-Series server deployment. All administrators should read this section to better understand the performance expectations. The *Compaq TaskSmart C-Series Servers Performance Guide*, document number 155E-0701A-WWEN, covers performance levels of the TaskSmart C-Series server series in depth.

Because networking variables can vary from network to network, performance will be different in each network. Higher cache hit ratios with smaller objects can increase the performance significantly. Request rates and response times are both affected by many variables, such as the number of objects per page, number of users, user bandwidth, the amount of time that a user views a page, latency and load on the origin server, cache hit ratio, and size of requested objects. With so many variables, it is difficult to predict how a TaskSmart C-Series server will respond in every environment.

Determining the correct size or model of a TaskSmart C-Series server for a network can be a complex process. We can simplify the selection by limiting the variables that we consider in making the selection. To choose the correct model, the only factor that needs to be known is the request rate. The request rate is the number of requests that the clients issue in a given second. The current request rate can be determined by analyzing the logs of an existing proxy. If there are no existing proxies, the request rate can be approximated using a simplified formula:

$$(\text{Number of users}) \times \frac{(\text{Number of web pages viewed in one minute by one user})}{(60 \text{ seconds})} \times \text{Number of objects on one web page} = \text{Requests per second}$$

Figure 25: Request rate formula

Measuring the Effectiveness of a Transparent Proxy

Since users generally do not call to report that the network is fast, positive feedback about the acceleration provided by the cache may not come readily. Some monitoring tools are able to measure response time and total retrieval time on a Web page. Administrators that are curious about the impact of the acceleration can install a response-time measuring tool. For network wide monitoring of the impact of the caching service, a robust package such as VitalSuite (www.vitalsigns.com) will install an agent on each client and report the response time to a master server. The network-monitoring packages are generally reserved for larger enterprises.

In order to measure the effectiveness of bandwidth savings, the cache-hit ratio provides insight to the bandwidth saved by the cache. For example, a cache that is maintaining a 28 percent cache hit ratio is filling 28 percent of the requests without consuming any bandwidth. The cache-monitoring panels in the TaskSmart C-Series server interface can provide the actual number of bytes that are saved by the cache. If a provider's charges are based on throughput, savings from the cache can easily be calculated.

Load Balancing and Load Distribution

This topic provides methods and mechanisms for increasing TaskSmart C-Series server performance with load balancing or load distribution. If a single TaskSmart C-Series server is not able to provide the performance necessary in the network, read this topic to learn what options are available for load balancing or load distributing transparent proxy requests in order to scale performance. If a single TaskSmart C-Series server can handle the loads of the network, skip to "Fault Tolerance."

Load Balancing

The performance of a TaskSmart C-Series server can be scaled nearly linearly by using an external load-balancing device. This can be done when the service request rate of the network exceeds the rated performance of the TaskSmart C-Series servers. It also can be done if the network traffic has grown to exceed what the current TaskSmart C-Series server deployment can effectively service. Two load-balanced TaskSmart C-Series servers can handle nearly twice the number of requests per second. Up to eight units, scaled using a Foundry ServerIron switch, have been tested in Compaq labs. This configuration produced over 90 percent scalability for load-balanced TaskSmart C-Series server performance. It should be emphasized that this performance scalability requires an external load-balancing device. The TaskSmart C-Series servers cannot distribute requests between themselves. Even the TaskSmart C-Series server clustering will not enable any level of load balancing between servers.

It should be emphasized that this performance scalability requires an external load-balancing device. However, for the transparent proxy deployment, the redirecting device most often will have the ability to load balance requests to more than one device. In many cases, as with the Foundry ServerIron switch, the load balancing in the L4, L7, or WCCP router is extremely effective and efficient. Most of the load-balancing algorithms, in the redirecting devices, are dynamic. They will ensure maximum scalability by evenly distributing the loads.

Load Distribution

Using DNS, round robin, or JavaScript proxy configurations are not an option for a transparent proxy. With a transparent proxy, the client browser no longer performs a DNS lookup or sends its request to a proxy. The interaction that causes the load distribution with DNS round robin and JavaScripted browser configurations never happens. An external mechanism is necessary to redirect and load-balance client requests to a transparent proxy.

As mentioned, load balancing provides nearly ideal performance scaling. Other performance related benefits can be realized by using a load-balancing device. Many of the load-balancing devices are also capable of connection or request limiting. While limiting connections or requests will not increase performance directly, these limits will provide insurance against overloading the TaskSmart C-Series servers. Rather than allow the TaskSmart C-Series server to become overrun and refuse connections, the load-balancing device may be capable of redirecting these connections to another TaskSmart C-Series server.

Fault Tolerance

This topic provides methods and mechanisms for increasing TaskSmart C-Series server fault tolerance with TaskSmart C-Series server clustering and external devices. If budgets or requirements do not mandate fault tolerance, skip to "Security."

There are two methods of ensuring system-level fault tolerance of TaskSmart C-Series servers:

- TaskSmart C-Series server software clustering
- External network device clustering

With the TaskSmart C-Series server deployed as a transparent proxy, there will generally be some redirecting mechanism. The redirector can be configured to handle system-level failure and maintain connectivity in the case of an L4, L7 or WCCP enabled router.

When an external device is performing the redirection, there is little need for the software clustering implementation because most redirecting devices can be configured to react to a system failure. Should the redirecting hardware discover that a TaskSmart C-Series server is no longer responding, the redirector could be configured to:

- Continue testing a downed server, allowing restoration.
- Continue redirecting requests to the remaining TaskSmart C-Series servers, if available.
- Cease the redirection function and forward all requests without the use of the TaskSmart C-Series server(s).
- Perform other feature specific to a manufacturer.

Switch manufacturers differ on the available fault-tolerance options. Check with the switch manufacturer to see if the desired configuration is available.

Make certain that the firewall and switch are both configured to allow client access in the event of a failure.

If available, use the failover features of the redirecting hardware. Software clustering is a failover feature of the TaskSmart C-Series servers. Caution should be used when considering this option, though software clustering can provide fault tolerance protection for a virtual IP address that provides cache services to clients or servers. However, this doesn't provide load-sensitive request distribution. Networking gear is often the only way to gain the shared capacity of multiple systems without a decrease in the total capacity.

Even though the TaskSmart C-Series servers already have a fault-tolerant software clustering mechanism, clustering for TaskSmart C-Series servers only implies fault tolerance. Keep in mind that clustered TaskSmart C-Series servers do not increase performance without an external load-balancing device. The clustering provides system level fault tolerance of all proxy services. Should one member of a cluster fail or stop accepting requests, the other members of the cluster will assume the addresses and the responsibility for the proxy services on that address.

In any environment where the TaskSmart C-Series server is required for client connectivity, fault tolerance should be seriously considered. Such is the case where the firewall has been configured to grant access only to the TaskSmart C-Series server. Clients would not be able to access any resources that lie outside the firewall if the proxy were unavailable.

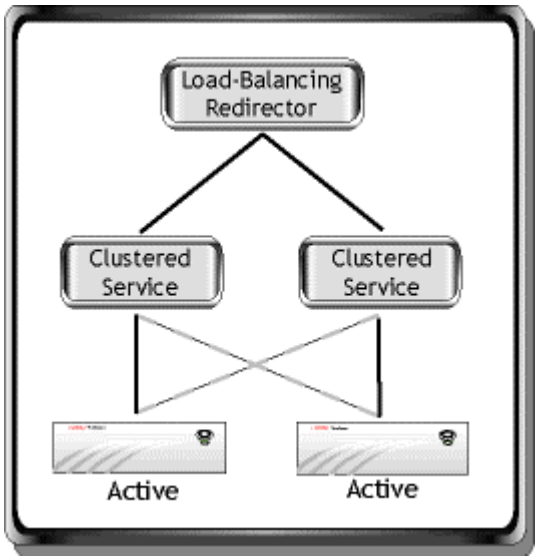

Figure 26: Load-balanced cluster

Only the servers that have virtual proxy service addresses assigned to them would handle requests. Therefore, if there are six TaskSmart C-Series servers and four virtual IP addresses, only four of the TaskSmart C-Series servers will be assigned any responsibility. The remaining two TaskSmart C-Series servers are awaiting a failure before they assume any responsibility.

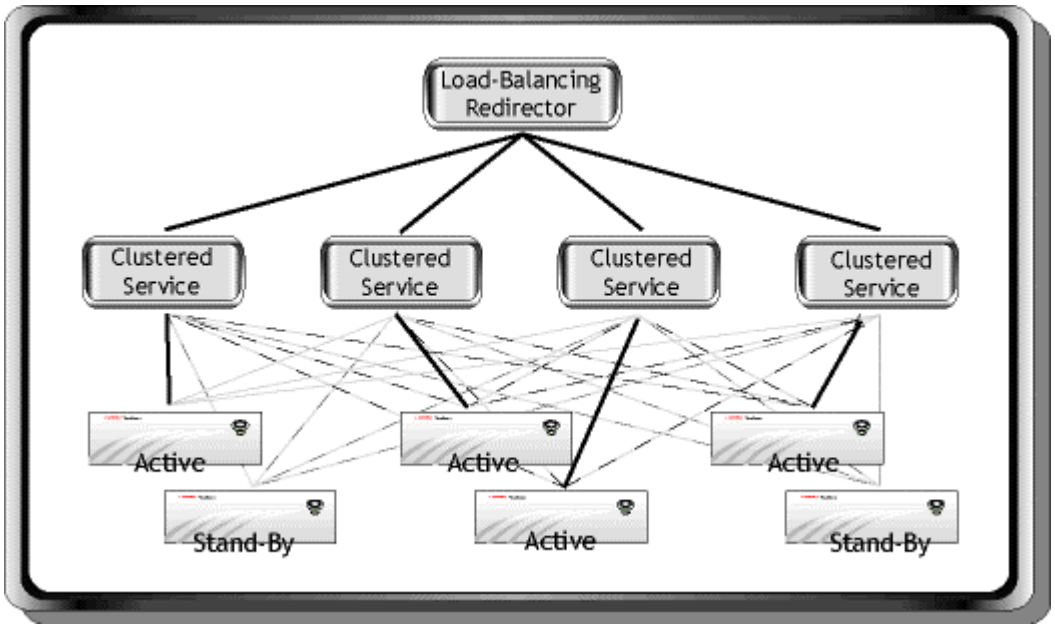

Figure 27: Load-balanced cluster with stand-by members

Security

A few areas draw security questions with the TaskSmart C-Series server. The most common question is in regard to secure client data. As mentioned, the TaskSmart C-Series server caching does not interpret this traffic. A properly deployed TaskSmart C-Series server will not interfere with or lessen the security of HTTPS traffic.

As far as security of the other objects in cache, only information that is publicly available on the Internet will be in cache. There is no way to access an object through cache that is otherwise not freely available on the Internet.

The only information that should be considered sensitive is the actual configuration of the TaskSmart C-Series server. The TaskSmart C-Series server provides for password protection on the GUI and Telnet configuration consoles.

When deploying a TaskSmart C-Series server, use the practice of taking all available security options. Make sure to add passwords to the configuration utilities, and Telnet. Only configure the minimum number of administration accounts on the system. User configuration on the TaskSmart C-Series server is only necessary for maintenance and configuration, not for the use of the proxy. IP address administration is also a powerful tool to restrict administration to only necessary IP addresses.

Even with secure password logon to the configuration utilities and restricted IP lists for administration, Compaq does not recommend deploying the TaskSmart C-Series server outside of the firewall. Previously, this guide mentioned reasons for performance and proxy services, but there are security issues as well.

When deploying the TaskSmart C-Series server within a firewall with access restrictions, it may be necessary to have the TaskSmart C-Series server use a user name and/or a password to access through the firewall. TaskSmart C-Series servers support access to firewall via SOCKS. If the firewall does not support SOCKS, then the firewall should be configured to restrict external access based on IP address.

LDAP can also provide user access control to prevent certain users from accessing the Internet. You must have a security system that can support the LDAP protocol or a security protocol gateway to the native security system. If necessary, certain sites can be unrestricted for all users.

If a current security system is in place, consider that only the standard HTTP protocol is supported between these devices. Therefore, place these systems in such a manner to allow all the required functionality while gaining the benefits of each of the systems.

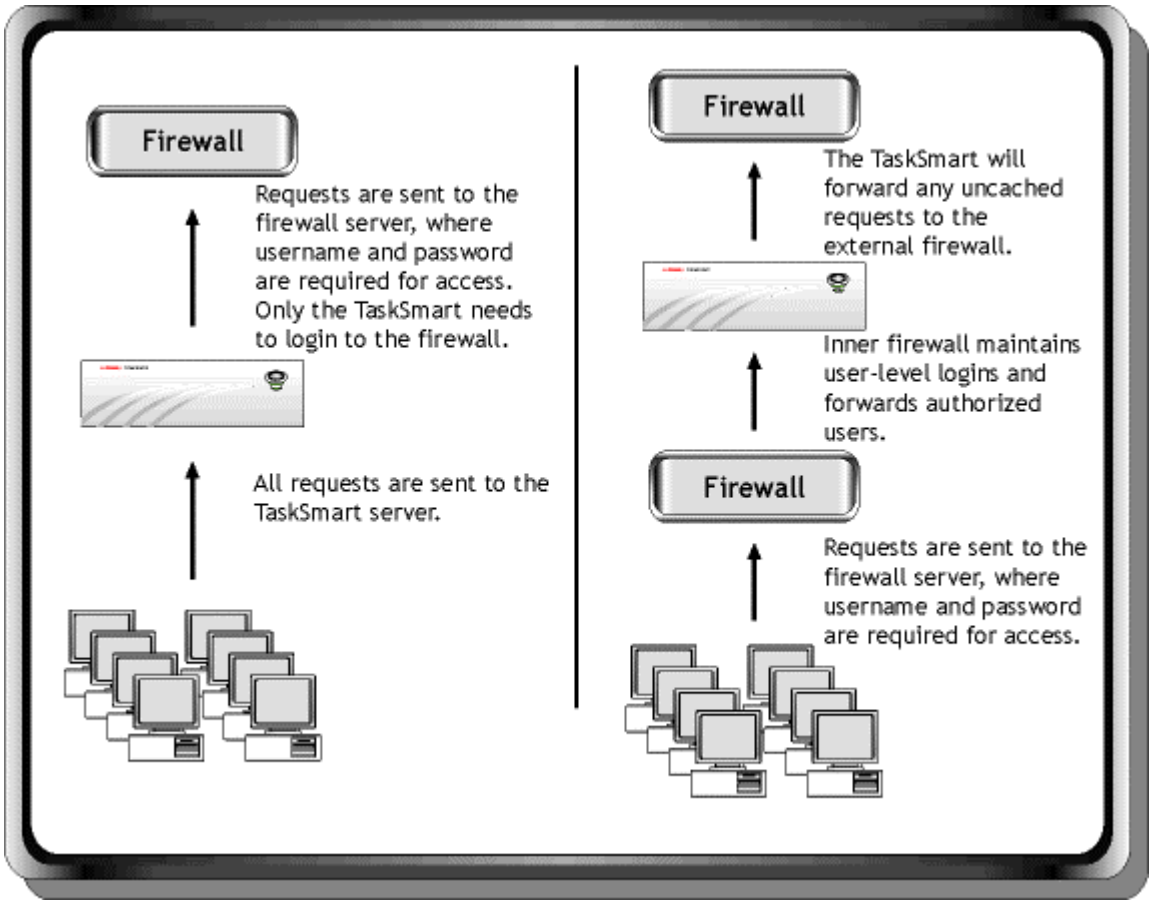

Figure 28: Controlling access with a DMZ

Conclusion

If network requirements mandate a transparent proxy deployment, this section should have answered most of the questions that determine the best location and configuration of the transparent proxy. Of course, some networks are too complex or unique to be considered in detail. Fortunately, the guidelines and considerations for deploying a TaskSmart C-Series server in simple networks also apply to networks that are more complex. If administrators and architects extend the guidelines and follow best practices, the TaskSmart C-Series server can be deployed as a transparent proxy in any environment.

Server Acceleration

In this section, the application of the TaskSmart C-Series server will be applied to a point of publishing. Server acceleration seeks to improve Web server scalability and performance by offloading redundant requests and HTTP connection management. This section explains the operation and deployment of the TaskSmart C-Series server. If the network is not a point of publishing or is dominated by client access, read the previous sections on "Client Acceleration."

Introduction

While caching is often associated with accelerated client access, caching can have an equally dramatic impact on servers as well. The technology and theory of server acceleration is the same as that of client acceleration. Frequently requested data is efficiently delivered to the clients and served without interaction from the origin server.

Consider for a moment the **INDEX.HTML** and all embedded objects. It is estimated that the **INDEX.HTML** file is served as much as twenty times more often than any other page on a website. In terms of server load, the greatest amount of resources is spent serving the same information repeatedly. Consider the same scenario with a TaskSmart C-Series server between the Web server and the Internet. All static objects referenced in the **INDEX.HTML** and the **INDEX.HTML** itself, if it is static, are now delivered to the clients from the TaskSmart C-Series server. Caching, however, is not limited only to the **INDEX.HTML**. Requests for all static objects on a Web server can be serviced from the TaskSmart C-Series server, which eliminates or reduces the load on the origin server.

These benefits are not limited to pages with the **.HTML** file extension. An even greater benefit can be obtained by caching objects that are dynamically created. Sometimes the easiest solution is just to add more Web servers. Simply adding another Web server might outweigh the cost of an engineer's time. We will explore techniques for off loading the origin server. These are advanced topics, requiring investigation of the cacheable content to be delivered. The techniques are intended as solutions for the most demanding and challenging hosting systems. If the cost of backend systems is high, then these techniques might provide the answer.

Off-loading The Origin Server

The power of new Web server scripting languages has changed the functionality of Web pages forever. Active server pages are powerful tools. They are used to personalize and empower the content available through Web servers, far beyond the original static information stores of yesterday's Web. These powerful techniques are not without a cost. Often these pages include database lookups and other backend accesses. Mainframe searches are even invoked to get the data required to create the pages desired by end users.

These systems are often sought by companies to allow them to compete in the new market created by the Internet. They allow companies to save millions by allowing customers to search databases traditionally only accessible by customer support or sales representatives. Potential customers can search for prices and products as well as technical data. This makes the company embracing these solutions much more competitive in today's economy.

End users expect Web pages to instantly respond and do not appreciate the incredible complexity invoked by simply clicking on a link. Few understand the entire system. Inefficiencies result from many separate parties, independently managing a large collection of interconnected networks and systems. Key individuals may be responsible for resolving issues that are created by the climate of the Internet. This is done with only having control of a few parts of the entire system. Scaling also tends to be an insurmountable task with an unending appetite for capital dollars. Companies will at times spend millions on upgrades only to be disappointed by end-to-end performance.

Content developers are often not aware of the inefficiencies created by the approaches they adopt. Management divides responsibilities for development and operation into different groups. Sometimes even different companies are responsible for operation verses development. Content developers test the code with the best methods they have, but these approaches do not effectively indicate the real world performance of the end-to-end solution. Content creation and aggregation from separate parties also contributes to the problem. A powerful and successful solution is the caching proxy. Reverse proxy Web caching can often provide the solution to the entire problem.

The concept is simple. Let us use the weather as an example. If someone looks up the weather in Chicago on the Web, then a complex and expensive system might be used to create the report for that user. If someone else looks up the weather for Chicago a few seconds later, the entire process is repeated. If this is happening dozens of times per minute, one can see the waste and futility of preparing a unique response for each end user. Caching requests for up to five minutes might be a reasonable solution. Hundreds of users might share the same response before the expiration of the content. The Web server's load is reduced while the backend systems are also offloaded. The reverse proxy enables all the end users' requests to be aggregated in the most logical place. Placing the cache engine just in front of the Web server allows the largest group of end users' requests to be served from the cache.

It might be completely reasonable to cache these pages for five minutes. Depending on the popularity of the system and the page being cached, five minutes may make a tremendous difference in the number of dynamic pages that must be generated by the origin servers. Other types of dynamic data might be cacheable for even an hour before it is expired. Images that make up your Web site might easily be cached for a day. Other static content such as Java Script and Style Sheets are examples of files that can be effectively cached for extended periods with great benefit to the Web-hosting company and the end users. Cache hit ratios have been observed in excess of ninety percent by these techniques.

To the Web server there seems to be only one client, the TaskSmart C-Series server. A typical enterprise Web server may have to manage thousands of HTTP connections. These connections also tax the performance of the Web server by consuming small, allocated blocks of memory and processor time. This is also true when the connections are idle. If a Web server directly serves the end-user clients, then it must have connections open with persistent sockets to all the end users that are using the website. As users stop browsing pages, the number of concurrent connections open will decline. If the TaskSmart C-Series server is deployed, then it can handle these connections for the end users and reduce the load on the Web server.

The unpredictable performance of the connection through the Internet to the end user's browser can impact the reliability of a website. A fiber cut on the nation's long distance network might degrade a connection between two or more parts of the Internet. Though the network is resilient and other connections handle the routing, packet loss may increase for users on some systems where a normally well-functioning link now experiences packet loss. Carriers tend to avoid unnecessary upgrades on their networks and cross connections. The connection's real cost links closely with the carrier's profit. If the packet loss suddenly increases for a small group of the users on a website, the number of connections required to serve data to all the end users of the website increases dramatically. A website that does not have a large reserve capacity of TCP sockets and the management capacity to ensure good performance on all the TCP sockets can suffer an outage. This can result from a simple routing degradation in the Internet.

Connection Offloading

Consider the time necessary to transfer a large HTML page to the cache. Fifty thousand bytes might take three tenths of a second at LAN transport speeds. Now consider the time necessary to transfer that same data to a client on the Internet with a 56-K modem. Fifty thousand bytes might take eight seconds to transfer at 56-K bits transport speed. If the TaskSmart C-Series server is used to handling the connections to the end users, then the connections to the server can be closed quickly. The TaskSmart C-Series server sends the response to the end user and frees the server to respond to another request. This benefit is realized regardless of whether a cache hit or a cache miss is the result of the request from the end user. This also dramatically increases the number of useable active server page (ASP) sessions on the Web server by reducing the time duration required to deliver the responses from the origin server.

Pinning Content

Pinning is the process of specifically configuring the cache engine to cache certain pages. This process overrides the natural behavior dictated by the HTTP headers and default behavior of the TaskSmart C-Series server. The use of pinning in the reverse proxy application, assuming that care is taken to confirm that specific objects are cacheable, can be an extremely effective measure to offload backend systems and reduce requests from even reaching the origin server.

With regard to streaming, popular video and audio streaming formats can be cached by streaming media capable units. Both live and on-demand streams can be supported. It is important to understand that the streaming origin server must be available to the cache appliance when serving these audio and video streaming formats. This is to allow the cache to receive the license authorization to serve the stream from the origin streaming media server. The cache can split the live stream and cache the on-demand streams. When the cache needs to serve streaming media, the cache must receive license authorization to serve any clients from the streaming origin server. If an origin server is licensed for twenty concurrent streams, the cache will not increase the concurrent capacity of the total streams to be served.

On a heavily loaded website, caching objects for even thirty seconds can greatly reduce the load on the origin servers and the backend data servers. Great care must be taken, however, the benefits are also very dramatic. All these techniques can make scaling a website through caching a very worthwhile endeavor, though one must work closely with the content developers to confirm that specific pages can be cached.

Some content creation methods run in direct opposition to caching. Methods of tracking end users' behavior that include adding a User Identification (UID) to the request string sent from the browser obfuscate the cache ability of base content that could be cached. This minimizes or effectively nulls all caching for all caches between the browser and the origin server. The result may include dramatically reducing the performance to the end user while causing every request to result in a connection to the origin server. Every page results in massive processing by all interconnected caches and servers while the end user waits for the page due to the resulting additive latencies.

Even in situations where content can't be cached for various reasons, cache configurations can still improve end-to-end performance by overcoming the additive effects of distributed packet loss. The speed of a TCP flow degrades exponentially from total packet loss end-to-end. Multiple distributed caches can reduce packet loss to levels where content can be retrieved in reasonable times when previous direct access performance was unacceptably slow.

Cache Headers

HTTP headers control the behavior of standards-based caches. Expired headers can be used to directly express the time objects are to be considered cacheable. The directive of the public will increase the number of forward caches that cache the objects. The **No-Cache** header controls the caches as expected. There are several other headers that can improve the performance of a website by allowing caching of static content by browsers and caches. The server **date** record and the **last-modified** record will further indicate caching opportunities for both the browser and the inline HTTP caches. The **expires** record can also be employed. Depending on the configuration of the origin server, these headers may or may not be included in the headers served by a particular website. Proper use can allow end users the fastest response with only the necessary updates sent to the end users' browsers.

Sometimes content developers add random numbers to minimize caching without understanding the industry-standard methods to prevent the caches from caching content. .ASP pages automatically are tagged as **No-Cache** from the server but developers can include code that changes this behavior. Few content developers set the appropriate headers on their content. The benefits of faster response time for end users and reduced bandwidth required for hosting the site are overcome by the developer's pressure to develop compelling and full-featured content.

A simple approach to prevent any stale cached image from being served to an end user is to change the name of any updated image. This allows aggressive settings for cache headers without any worry if an update is required. Caches handle the removal of unneeded content without any administrator intervention. For more information see "Cacheable Data,' "Cache Control," as well as the references at the end of this document. In addition, HTTP 1.1 and RFC-2616 provides a complete list of headers and intended behaviors. Further details can be found on the **cache.config** file in the *Compaq TaskSmart C-Series Administration Guide*.

Security via Secure Sockets Layer (SSL)

When an end user connects to a secure site, all accesses and responses are encrypted. The requests are made from the browser with the HTTPS protocol instead of HTTP. Interconnected caches just forward these encrypted requests to the origin server and reply with the encrypted responses.

If a browser switches to HTTPS from HTTP, depending on the settings, a notice will be displayed. End users feel uncomfortable using sites that contain mixed security of both HTTP and HTTPS. For this reason companies often secure the entire site with SSL using HTTPS. However, securing the content of a website via SSL reduces caching effectiveness.

A solution to allowing caching and, therefore, to speeding users requests is as follows. Place all public data in HTTP://WWW.domainname.com so that data is cached by both edge forward proxies and, if present, reverse proxies. When the end user needs to access secure areas of data, another HTTPS server running on the same system with the name HTTPS://Secure.domainname.com could serve the secure content.

SSL session setup and processing is cryptographically complex and processor intensive. For this reason, placing SSL accelerators in front of caches at secure sites allows the connection between the end user and the hosting company to be encrypted while the internal traffic from the SSL accelerators and the caches can be served via HTTP. In this way, reverse proxies can be used internally to offload Web servers. This also relieves the Web servers from the challenge of encrypting and decrypting the data passing between the servers and the end users. The Compaq TaskSmart line includes SSL accelerators.

Methods of Deployment

For a server accelerator, unlike a client accelerator, there is only one primary mode of deployment: reverse proxy. A reverse proxy replaces the Web server as the publishing node for a website. Clients submit requests across the Internet to the IP address of the TaskSmart C-Series server. The TaskSmart C-Series server examines each request and responds to those requests that can be filled from cache. It then forwards, to the origin server, those requests that cannot be filled from the internal cache objects stored.

As a reverse proxy, the TaskSmart C-Series server will rely on two channels of communication. One path is to the clients and the other is to the origin server. In order for the TaskSmart C-Series server to receive the client requests, there must be a mechanism to direct the clients to the TaskSmart C-Series server instead of the origin Web server. By matching the TaskSmart C-Series server IP address to the DNS entry of the website, client requests will be issued to the TaskSmart C-Series server. Correlating the TaskSmart C-Series server IP to the DNS entry can be accomplished by either deploying the TaskSmart C-Series server at the IP address of the Web server or by modifying the DNS records to point to the TaskSmart C-Series server instead of the Web server.

When changing the IP address of the origin Web server, it is not necessary to assign a public address. Instead, the Web server itself may exist on a local non-routable subnet. In this case, one of the other interfaces on the TaskSmart C-Series server would then have to be placed on the same non-routable subnet. With the origin server on a non-routable subnet, remote and local clients are unable to access the Web server directly, which makes the Web server impervious to attack or security breaches.

Reverse Proxy

This section of the TaskSmart C-Series Server Deployment Guide provides all the information that is required to make an effective deployment of a TaskSmart C-Series server as a reverse proxy server accelerator. In this section, all relevant details about operation, deployment, and architecture are presented. The information contained applies to all environments, from smaller single-server websites to the larger enterprise-class Web presence.

Introduction

The application of a reverse proxy is quickly becoming one of the most effective methods of managing increasing loads and ever-expanding Web server farms. While this application of the TaskSmart C-Series server has different goals and guidelines than a client accelerating proxy, the technology is the same. The mission is to serve redundant data quickly from the cache reducing the load on origin servers.

The benefits and goals for a reverse proxy can be quite different from either the forward proxy or transparent proxy client accelerators. Generally, three primary benefits can be realized by using the TaskSmart C-Series server as a reverse proxy/server accelerator:

- Bandwidth savings are realized because redundant downloads from the origin server can be reduced.
- Improved response time of a website can be accomplished by taking advantage of the streamlined HTTP engine in the TaskSmart C-Series server.
- HTTP connection management is realized by allowing the TaskSmart C-Series server to receive all client connections reducing the load on the origin server's protocol stack.

Because a reverse proxy will generally be farther away from the clients, the potential bandwidth savings are realized between the reverse proxy and the Web server. In some cases, placing the TaskSmart C-Series server upstream at the provider can eliminate traffic from the WAN link and provide greater connectivity. Because the provider will generally have greater connectivity than the customers will, placing the point of publishing at the Internet transit provider removes the leased-line bottleneck. Sensitive information can be retained at the local end of the line. Keeping the Web server locally can reduce security issues and reduce liability.

In many aspects, a reverse proxy is the most straightforward deployment. As a server accelerator there is little practical need for proxy services such as access control, filtering, or hierarchies. The "redirection" mechanism that directs client requests to the TaskSmart C-Series server, as opposed to the origin Web server, is already in place with DNS. Finally, there is no proxy configuration required on either the clients or the origin Web server.

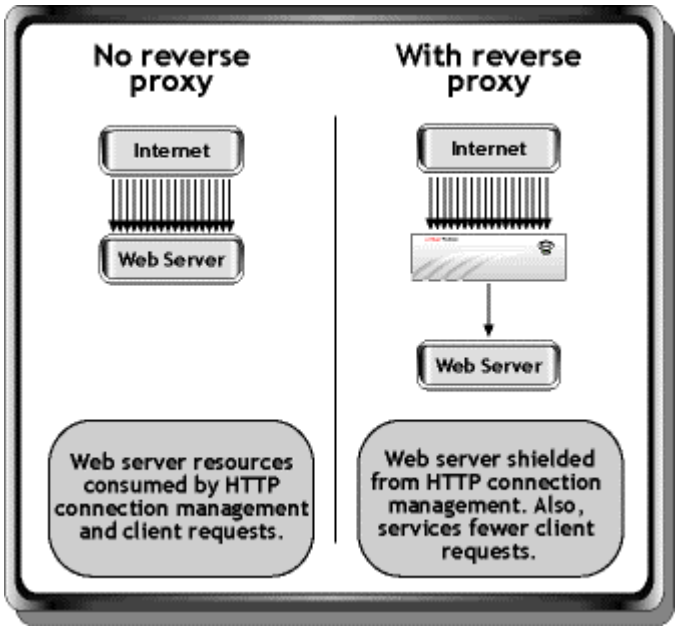

Figure 29: HTTP connection management of a reverse proxy

Reverse Proxy Operation

When deployed as a reverse proxy, the TaskSmart C-Series server becomes the point of publishing for a website. To all nodes, the TaskSmart C-Series server seems to be the Web server. Only the TaskSmart C-Series server needs to know the actual name or address of the origin server.

Clients on the intranet and Internet will perform DNS lookups and should get the address of the TaskSmart C-Series server as the reply. Clients will issue the request and connect to the TaskSmart C-Series server. Requests that can be filled from cache will immediately be sent to the client. Requests that are not already in cache or cannot be held in cache are reissued to the origin server. The origin Web server would then respond to the TaskSmart C-Series server's request. The TaskSmart C-Series server would then cache any cacheable objects and forward the objects to the client. To clients on the Internet there is no indication that the request was filled by the TaskSmart C-Series server. To the origin Web server, it seems as though the TaskSmart C-Series server was the only client that issued a request.

With reverse proxy enabled, the TaskSmart C-Series server will listen and service HTTP requests on a definable port. The port is almost always port 80 for a reverse proxy. Any non-HTTP requests are forwarded to the origin server for processing. In this manner, the TaskSmart C-Series server will not interfere with any non-HTTP services that the Web server is providing.

The TaskSmart C-Series server can also function as an accelerator for a multi-homed Web server. Individual configuration is required to accelerate multiple websites from a single server. This introduces some practical limits when doing configuration through the Web browser interface. If there are a large number of sites that will be accelerated by a single TaskSmart C-Series server, it may be practical to learn the appropriate file configuration details. Refer to the *Compaq TaskSmart C-Series Administration Guide*.

Cacheable Data

For a reverse proxy, some types of traffic are not cached even if they are HTTP requests. The TaskSmart C-Series server will not cache objects that are marked in the HTTP header as non-cacheable. Nor will it cache objects that carry implied non-cache ability.

Since the introduction of HTTP, there has been the ability to control intermediary caches from the origin server. Originally, this was done with the HTML header "PROGMA NO-CACHE." HTTP 1.1 extended the definitions of cacheable data to include expirations. Additional information on cacheable objects can be found in the *Compaq TaskSmart C-Series Administration Guide*, Chapter 3.

Three types of objects that are not cached:

- Objects that are marked non-cacheable in the HTTP header
- Objects exceeding the maximum object size and are a configurable variable on the TaskSmart C-Series server
- Objects that are implied as non-cacheable

Table 4. Cacheable Data

Type of Data	Is it Cached?
HTTP	Yes, unless marked otherwise by origin server or exceeds maximum file size for cacheable objects.
HTTPS	No
Password-protected FTP	Yes, but only served from cache to authenticated user.
Anonymous FTP	Yes, unless it exceeds maximum file size for cacheable objects.
Password-protected FTP	No
.ASP	No, requires server-side processing. Static-embedded objects are cached.
.CGI	No, requires server-side processing. Static-embedded objects are cached.
Cookies	No, origin server should mark cookies as "PRIVATE."

Not all non-cacheable HTML is actually marked as non-cacheable in the HTTP header. Implied non-cacheable data is the case on any HTML that requires server-side processing, such as .CGI or .ASP. This does not mean that the TaskSmart C-Series server will not accelerate .CGI or .ASP pages, only that the origin Web server will still be involved in generating the scripted HTML. TaskSmart C-Series server caching is done at the object level, not the page level. This means that any static images (.JPG, .BMP, .GIF, and so on) are all cached. In a given .ASP-generated HTML document, there may be many static objects that remain the same each time the page is viewed. These objects can be cached. A corporate logo for instance would be static and cacheable. These objects will be cached and when they are next requested as part of a dynamically generated HTML document they will be served from the high-speed local cache.

In terms of bandwidth, the embedded objects and images that are static and cacheable represent a considerably larger percentage of the total size of a Web page. Even in dynamic pages that are generated at request time, the majority of the data may be cacheable.

Another type of data that is not cached is HTTPS or secure data. In fact, HTTPS data is simply passed through the TaskSmart C-Series server. Any banking transactions or online purchases are essentially transparent to the TaskSmart C-Series server. Since HTTPS uses port 443, the proxy services are not applied to secure sessions. Any non-secure items that are embedded in secure pages are completely cacheable. Examples such as logos and buttons at the top of a bank statement are not sensitive data and, therefore, do not need to be secure and may be cached.

Secure data should not be confused with private or password-protected data. Data that is static and requires a user name and password to be accessed is cached. If a user name and password are required to retrieve the objects from the origin server, then the objects are marked so that only the authenticated user can view them from the cache. They are cached only for the single user. If another user uses a different logon to view the same information, the information will not yet be cached for the second user. The user name and password are not cached and the authentication is still performed on the origin server.

Cookies are another type of object that may not be cacheable. The origin server should mark any cookies that contain user-specific data as non-cacheable or "private." Any objects that are marked "private" cannot be held in any cache other than the client browser. If the origin server does not indicate that a particular cookie is private or non-cacheable, then the TaskSmart C-Series server would hold the cookie in cache and respond with the cookie information for future requests.

Some objects may be cached for only short periods. In these cases, an origin server may specify an expiration time for an object in the HTTP header. When the expiration time has passed, the TaskSmart C-Series server must retrieve the object from the origin server and discard the one held in cache when the object is next requested. The TaskSmart C-Series server is not allowed (by the HTTP specifications) to vend out objects that it believes may be expired. This means that if an object has expired and the origin server is not available, the TaskSmart C-Series server will not display the expired data.

Cache Control

With the advent of HTTP 1.1, Web masters and Web authors were given enhanced control over caches that handle their data. Some of the cache-control headers that are defined in the HTTP 1.1 specifications are as follows:

- No-cache
- Public
- Private
- No-store
- Must-revalidate
- Proxy-revalidate

These HTTP headers can be implemented in the **<HEAD>** section of the HTML document. Generally, it should be implemented in the actual HTTP header definition on the origin server rather than the parsing of HTML. If the meta tags are defined in the HTML, as shown in the following example, it is likely that only the client Web browser will recognize and adhere to the cache control directives.

```
<HTML>
<HEAD>
<META HTTP-EQUIV=”CACHE-CONTROL” CONTENT=”PRIVATE”>
<META HTTP-EQUIV=”PRAGMA” CONTENT=”NO-CACHE”>
<META HTTP-EQUIV=”EXPIRES” CONTENT=”0”>
<TITLE>
The TaskSmart C-Series server enables new levels of Web server performance
</TITLE>
</HEAD>
<BODY>
```

Configure the Web server to generate the appropriate header information before the page is delivered. This will ensure that the TaskSmart C-Series server and all other caching proxies between the origin Web server and the client are adhering to the HTTP cache control headers. HTTP headers may be generated by .CGI, .ASP scripts, in Apache, and CERN. For Microsoft IIS, use the Microsoft Management Console for IIS to view the **HTTP Header Property Sheet**. This can be used to set the cache control value returned to the browser in the header of the HTML page. For example:

```
Response.CacheControl = Private
Response.Pragma = No-Cache
Response.Expires = 0
```

With the TaskSmart C-Series server handling every request for a given domain or set of domains, proper use of cache control directives become more critical to ensuring that clients are always receiving the most up-to-date data. It is important to note and understand the way in which the origin Web server application implements the cache control and expire directives. Administrators should spend some time investigating and generating the HTTP headers on their chosen Web server platform. Details on the **cache.com** file can be found in the *Compaq TaskSmart C-Series Administration Guide*.

Architecture and Placement in a Network

Because the TaskSmart C-Series server will replace the origin Web server as the point of publishing for an accelerated website, the TaskSmart C-Series server should be placed between the Web server and the Internet. As a starting point, place the TaskSmart C-Series server at the same address(es) and logical location that the Web server would normally be placed. From there, try to reduce both the number of hops between the TaskSmart C-Series server and the Internet, and the number of hops between the TaskSmart C-Series server and the origin server.

- Place the TaskSmart C-Series server between the Internet and the Web server. To receive client requests as the Web server and issue requests to the Web server as a client, the TaskSmart server should be positioned logically between the two. This is a proxy and therefore it will act as an intermediary between two nodes. Network placement should reflect and facilitate this functionality.
- Place the TaskSmart C-Series server as close to the Internet as possible. Maximize potential throughput and minimize potential bottlenecks by deploying the TaskSmart C-Series server as close to the Internet as possible. Generally, points within the Internetwork that are closer to the backbone have greater throughput and lower latency. Take advantage of the TaskSmart C-Series server performance by placing it as close to the access point with the greatest bandwidth.

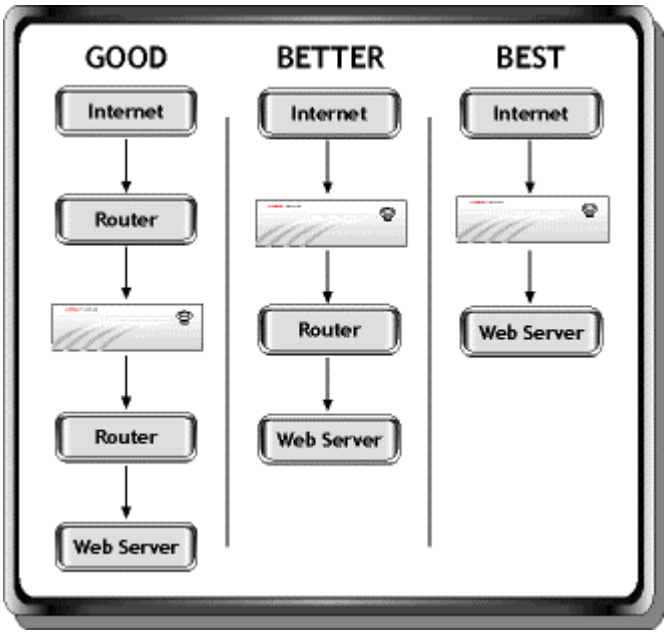

Figure 30: Minimizing hops and placing close to the Internet

- Place the Web server as close to the TaskSmart C-Series server as possible. For requests that must be filled by the proxy from the origin server, reduce potential latencies and reduce network congestion.

Unlike the client accelerators, there should be no need to implement any proxy hierarchies. The TaskSmart C-Series server proxy services of caching and logging are generally all that would be applied to a reverse proxy. There is little need to filter inappropriate content from one's own Web server. Since there is no ubiquitous user database for all users on the Internet, access control is not available. This does not mean that any Web server-based authentication will not work properly. Any Web pages, directories, or objects that are password protected on the origin server will still require a user name and password to view. Authentication and access control are not the same. Access control would apply filters or permissions to a user based on information stored in a user database such as a directory or a domain. With this in mind, the access control on an external website would be nearly impossible.

Any internal servers that use Windows NT Challenge/Response (NTLM) authentication cannot be accessed through a reverse proxy. NTLM does not allow any intermediaries to proxy requests between the client and the server. This means that the Web server cannot reside behind a reverse proxy and the client may not access the Web server through a client accelerating forward or transparent proxy.

Physical Connections

Questions may remain concerning implementation of the architecture through physical connections. A requirement of the physical connections is that they must provide stable connections to all necessary resources. The TaskSmart C-Series server must be able to reach the default gateway for proper operation. All other network resources must be accessible directly or through a network gateway. It must be on one or more of the physical connections. These resources can be clients, servers, and DNS services.

It is possible to use multiple interfaces in your configuration. Using the primary interface for internal connection and the second interface for public access is a common approach. Non-routable network addresses can increase security for internal networks and servers, while allowing the TaskSmart C-Series server to retrieve data from origin servers. The physical interfaces should provide the maximum bandwidth necessary for your configuration. If required, a Gigabit Ethernet NIC option is available. Refer to *Compaq TaskSmart C-Series Servers Performance Guide*, number 155E-0701A-WWEN, to review your sizing requirements and capabilities.

DNS Servers, Firewalls, and Routers

The TaskSmart C-Series server will replace the Web server as the point of publishing for all HTTP requests for the accelerated website. Any clients who submit requests to the accelerated domain should submit the request to the TaskSmart C-Series server. Configure DNS to resolve all client lookups for the accelerated domain to the TaskSmart C-Series server rather than the origin Web server.

The only location in which the name or address of the Web server must be displayed is in the TaskSmart C-Series server configuration. For security purposes, a reverse proxy allows administrators or architects to completely isolate users from directly accessing the Web server. This does not inhibit the serving of all intended content.

Firewalls generally do not require any reverse proxy specific configuration. Simply there are only two concerns that should suffice for all deployments. The first is making sure that clients are able to send requests through the firewall to the server. The second is that the server can respond. Should the TaskSmart C-Series server be deployed in a DMZ with the origin server lying behind the inner firewall, that firewall will have to be configured to allow the TaskSmart C-Series server and Web server to send requests and responses. To further increase security, the innermost firewall that separates the TaskSmart C-Series server and Web server could be configured to allow only the TaskSmart C-Series server to send requests to the Web server.

If the TaskSmart C-Series server were deployed in place of an existing Web server as a reverse proxy, there would be no changes necessary to the routing path for either incoming or outgoing requests. The default gateway for the TaskSmart C-Series server will point towards the clients, not the origin Web server. If the origin Web server is not on the same local subnet as the TaskSmart C-Series server and the default gateway is not able to reach the origin server, a static route must be used. A static route can be set in the TaskSmart C-Series server configuration for the address or subnet of the origin server. This may be done by using the TaskSmart C-Series System Administration Utility and clicking the **Linux Configuration>networking>routing** menu option. The same setup may be required for internal DNS to work properly in some scenarios as well.

When configuring a reverse proxy server accelerator, the administrator should supply to the TaskSmart C-Series server an IP address, DNS server address(es), a default gateway, and the address of the origin server. The TaskSmart C-Series server configuration is the only place that the origin Web server address needs to be known. No devices, other than the TaskSmart C-Series server, will make requests to the origin server.

Performance of a Reverse Proxy

The *Compaq TaskSmart C-Series Servers Performance Guide*, document number 155E-0701A-WWEN, covers performance levels of the TaskSmart C-Series server in depth.

From the Web server point of view there is only one client. There is considerably less work to be done in establishing and managing HTTP and TCP/IP connections. The LAN transport speeds allow content to move very quickly between the origin server and the cache. Even when being bombarded by thousands of users, the TaskSmart C-Series server will open only a handful of connections to the origin server. Regardless of the number of requests that the TaskSmart C-Series server is handling, any cache misses that need to be filled from the origin server will be done through the few persistent pipelined connections between the TaskSmart C-Series server and the origin server.

This buffering of connections is an effect of the reverse proxy and not dependent upon the cacheable property of the objects being served. Web servers that are serving purely dynamic data can still alleviate tremendous loads by deploying a TaskSmart C-Series server as a reverse proxy. The impact of this offloading can be measured instantaneously at the Web server by using performance monitors such as **MONITOR.NLM** on a NetWare server or **PerfMon** for Windows NT.

Some of the data will be cacheable and for each redundant request for a cacheable object, the TaskSmart C-Series server will fill the request without generating any traffic or load on the origin Web server. No website is purely dynamic. Even pages that seemingly never display the same image twice are usually drawing from a vast database of static images. When each of these objects has passed through the cache, even if they are part of dynamic HTML, the objects will be served from cache. For websites with a substantial amount of static information, the performance increases afforded by a TaskSmart C-Series server are unparalleled.

Load Balancing and Load Distribution

Client requests that are incoming to the TaskSmart C-Series server will require some external load balancing or distribution to span requests across multiple reverse proxies. Because nearly every request will involve a DNS lookup, DNS round robin is an obvious choice for distributing the requests. By having multiple records for the same domain name record, a DNS server can provide different addresses for alternating DNS requests. In this way, incoming client requests can be spanned across many servers with no extra investment and no extra configuration on the servers themselves.

DNS round robin has limitations to its load balancing ability. DNS round robin would more accurately be described as load distribution than load balancing. DNS round robin will not adaptively respond to a severely uneven distribution of requests. A true load balancer would recognize the disparity between the loads on each server and make any necessary adjustments.

With a true load balancer and multiple TaskSmart C-Series servers as reverse proxies, the IP of the load balancer becomes the sole published address for the website. Clients then issue the requests to the load balancer. The load balancer can distribute the requests to multiple machines to be serviced. Most IP load balancers, such as the Cisco Local Director, have configurable algorithms that can control exactly how the load balancer determines which machine will get a specific request.

In many enterprise environments, it may also be necessary to load balance requests that are being issued by the TaskSmart C-Series server to the origin servers. A load balancer placed between the TaskSmart C-Series server and the origin Web servers may provide the required control. The TaskSmart C-Series server itself is capable of distributing requests to the origin servers. Testing has shown that the load balancing in the TaskSmart C-Series server is not as efficient as a dedicated hardware load-balancing device. This should be kept in mind when designing an array of balanced servers.

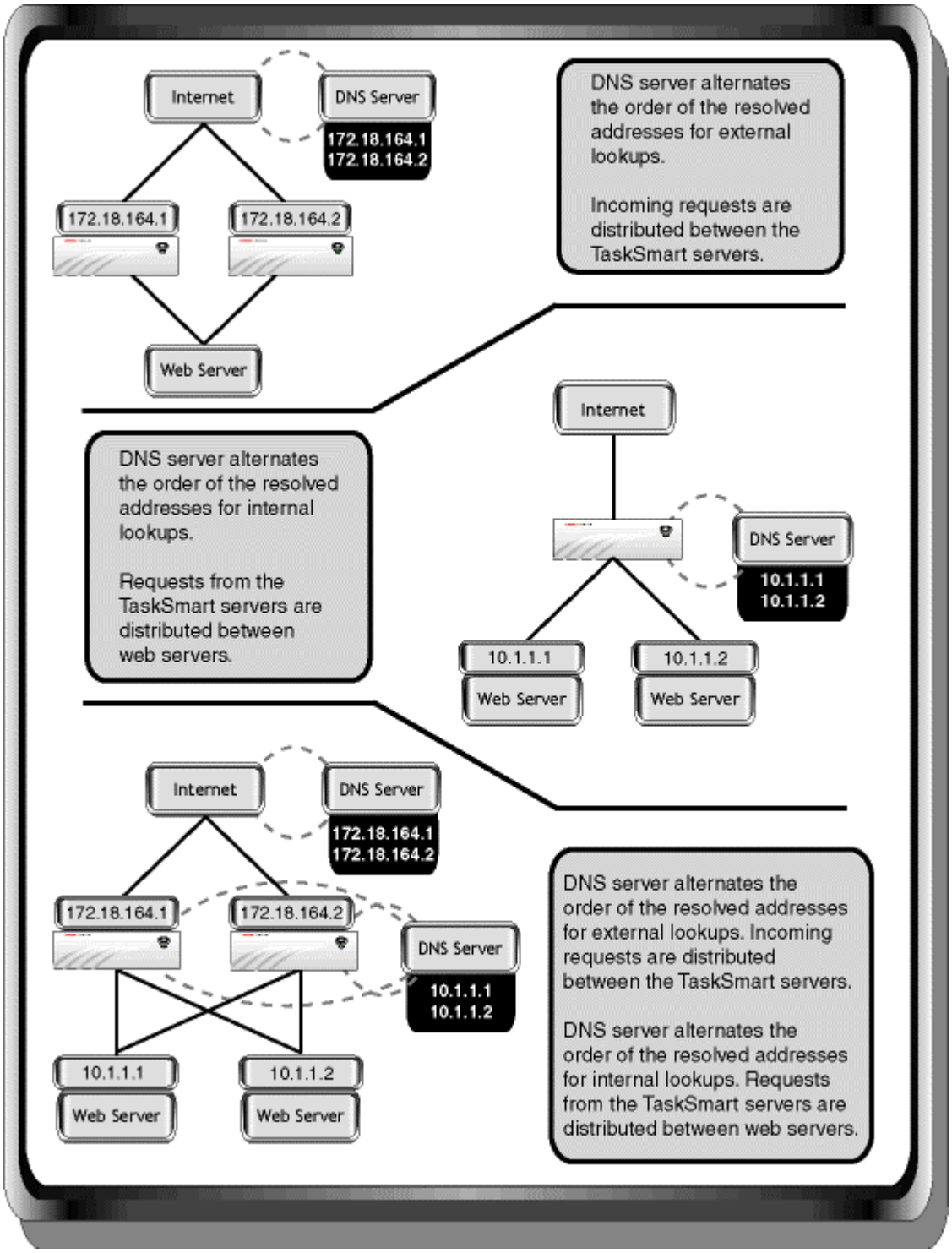

Figure 31: Load balancing using DNS round robin

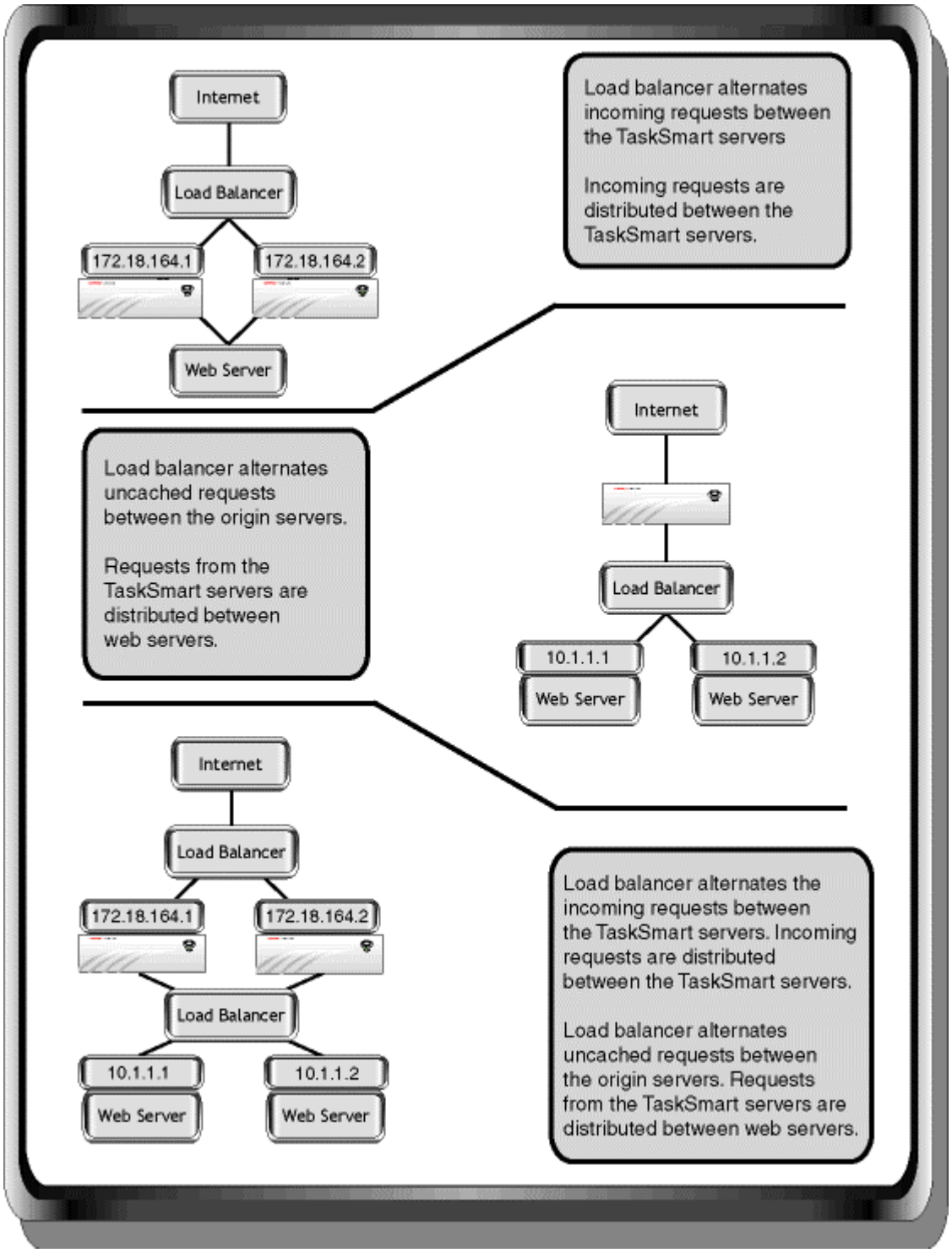

Figure 32: Load balancing using a load balancer

Fault Tolerance

When deploying an array of servers, fault tolerance for an Internet-enabled business is paramount. This topic provides methods and mechanisms for increasing TaskSmart C-Series server fault tolerance with TaskSmart C-Series server clustering and external devices. If budgets or requirements do not mandate fault tolerance, skip to "Security."

There are two methods of ensuring system-level fault tolerance of TaskSmart C-Series servers:

- Using the TaskSmart C-Series server clustering mechanism
- Using an external device

With the TaskSmart C-Series server deployed as a reverse proxy, clustering can provide system-level fault tolerance that would allow uninterrupted and automated failover in the event that a system became unavailable. Clustering for TaskSmart C-Series servers only implies fault tolerance. Keep in mind that clustered TaskSmart C-Series servers do not increase performance without an external load-balancing device, or some implementation of load distribution. The cost of load-balancing devices has declined drastically in recent years. Using an external device often provides the best combination of service fault tolerance, load sharing, and performance.

A benefit to clustering is a single copy of cached objects that are stored and then shared by all participating caches. This benefit is similar to that achieved by configuring multiple caches to participate as peers through ICP. The clustering provides system-level fault tolerance of all proxy services. Should one member of a cluster fail or stop accepting requests, another member of the cluster will assume the address and the responsibility for the proxy services on that address.

When using software clustering and multiple servers are handling a single-cluster service, one member of the cluster is responsible for all requests that are sent to a virtual IP address. All other members of the cluster are simply in a "stand-by" mode. They are not handling any traffic until the currently responsible server fails or is downed intentionally. Should there be a failure, the responsibility for requests sent to the virtual IP address would shift to one of the remaining members of the cluster.

Once again, load distribution is not achieved unless multiple virtual addresses are configured and clients are distributed to both addresses. An external load-balancing device is required to distribute requests between the members of the cluster.

When an external device is performing the load balancing, there is little need for the software clustering implementation because most load balancers can be configured to react to a system failure. Should a load balancer discover that a TaskSmart C-Series server was no longer responding, it could be configured to:

- Continue to check a downed server, allowing restoration.
- Continue redirecting requests to the remaining TaskSmart C-Series servers, if available.
- Bypass the load balancing to the TaskSmart C-Series servers and send all requests to the Web servers directly.

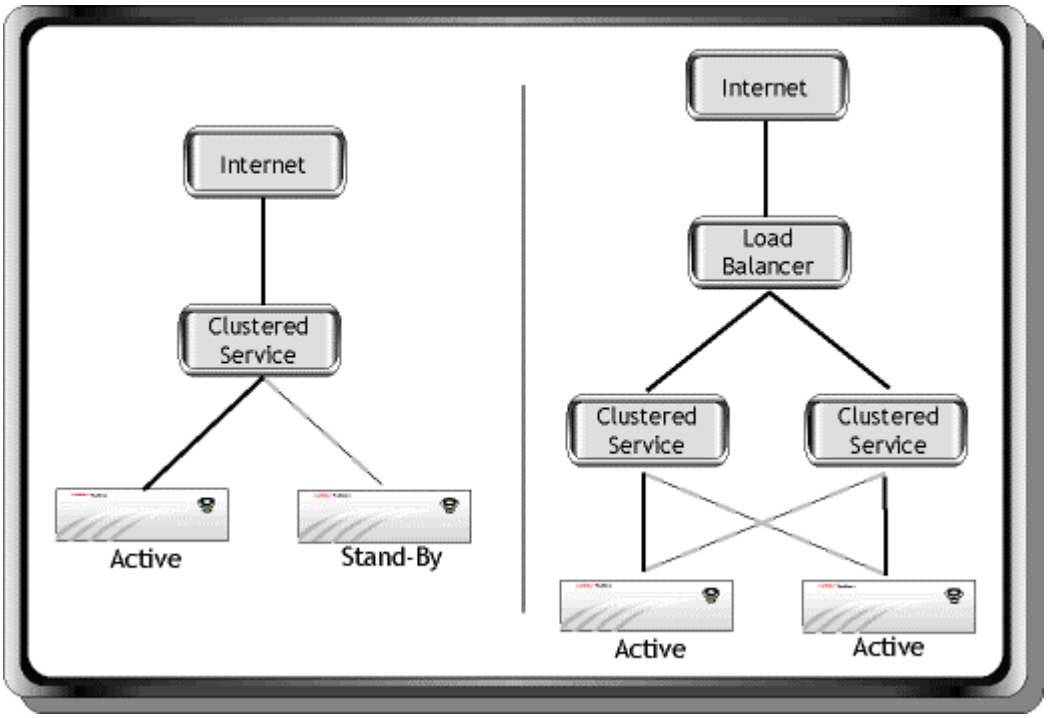

Figure 33: Reverse proxy cluster and load-balanced reverse proxy cluster

Manufacturers differ on the available fault-tolerance options. Check with the manufacturer to see if the desired configuration is available.

In any environment where the TaskSmart C-Series server is required for website operation, fault tolerance should be seriously considered. Deploying a cluster or multiple TaskSmart C-Series servers connected to a properly configured L4 switch will ensure that the proxy services are fault tolerant.

Security

The most common question is in regard to secure client data. The TaskSmart C-Series server caching service does not interpret this traffic. A properly deployed TaskSmart C-Series server will not interfere with or lessen the security of HTTPS traffic.

As far as security of the other objects in cache, only information that is publicly available on the Internet will be in cache. There is no way to access an object through cache that is otherwise not freely available on the Internet.

The only information that should be considered sensitive is the actual configuration of the TaskSmart C-Series server. For this reason, set the password protection on the GUI and Telnet configuration consoles. Be sure that the passwords you choose are secure and can't be guessed easily.

When deploying a TaskSmart C-Series server, use the practice of taking all available security precautions. Make sure to use strong passwords for all logon IDs. Add only a small number of user names to the system. The user names, that are administered through the GUI, are only required for the management and configuration of the system. This process is ICS based. For additional information, refer to *Compaq TaskSmart C-Series Servers Feature Production*, document number 158D-0701A-WWEN.

Even with secure password logon to the Management Utilities for administration, Compaq does not recommend deploying the TaskSmart C-Series server outside of the firewall. If a TaskSmart C-Series server were placed outside of the firewall with client acceleration enabled, any user on the Internet could use the proxy to accelerate their access or mask their address. Eventually, the users for whom the cache was intended would find that the external users are affecting access and performance.

When deploying the TaskSmart C-Series server within a firewall with access restrictions, it may be necessary to have the TaskSmart C-Series server use a SOCKS connection to access content on the Internet. Refer to the firewall and SOCKS manual for guidance.

As a reverse proxy, the TaskSmart C-Series server can actually add another layer of security to a Web server. With one network interface on a public subnet and the other on a non-routable subnet such as 10.1.1.X, the TaskSmart C-Series server could both receive client requests and fill any necessary requests from the origin server. Clients cannot reach the origin Web server itself. This does not require network address translation or routing by the TaskSmart C-Series server. The client request was destined for the TaskSmart C-Series server. When it arrives and the object is not in cache, the TaskSmart C-Series server, not the client, will retrieve the object from the origin Web server. The Web server cannot see the client and the client cannot see the Web server. Only the TaskSmart C-Series server can see both the client and the Web server. In effect, this can make a Web server extremely secure and impervious to any malicious clients. The TaskSmart C-Series server will log all HTTP requests. Depending on requirements, make plans to store these files for later review.

Conclusion

At this point the operation, placement, and architecture should be understood so that a successful deployment is possible in nearly every network. The deployment guidelines and principles presented herein apply to the most basic Web pages and the most massive e-commerce Web presence. While it is impossible to cover every aspect of the network architecture, hopefully this guide has answered the majority of the questions and reduced the deployment of the TaskSmart C-Series server to a truly simple procedure.

Where to Get More Information

Caching

www.caching.com/

www.web-caching.com/

Caching Tutorial for Web Authors and Webmasters
www.web-caching.com/mnot_tutorial/

Proxy auto-configuration file format

home.netscape.com/eng/mozilla/2.0/relnotes/demo/proxy-live.html

TaskSmart C-Series Servers

www.compaq.com/tasksmart/cseries

L4 Switches and Redirectors

www.foundry.com/

www.alteon.com/

www.f5.com/

www.radware.com/

www.cisco.com

WCCP and Cisco Routers

www.cisco.com/

Inktomi

www.Inktomi.com

Related Protocols

www.ietf.org/

www.rfc-editor.org

Squid Log Analysis Tools

www.squid-cache.org/Scripts/