

Reverse Proxy with SSL - ProxySG Technical Brief

What is Reverse Proxy with SSL?

The Blue Coat ProxySG includes the functionality for a robust and flexible reverse proxy solution. In addition to Web policy management; content filtering, blocking, and virus-scanning; organizations can implement a reverse proxy solution with SSL front-ending their Web applications for SSL security and greater Web performance.

An HTTPS “reverse” proxy sits on the edge of a corporate network and accepts requests from users on the WWW coming into a corporate website. It is typically used to offload SSL processing from the server to the proxy, cache server content, and optionally provide threat detection and data leak prevention checks. In a content delivery network (CDN) it can also be used for bandwidth management and server acceleration deployed in the middle of the network – away from the server but not necessarily at the branch or network edge. Communications between the HTTPS reverse proxy and the server might or might not use SSL.

ProxySG reverse SSL proxy provides the following advantages:

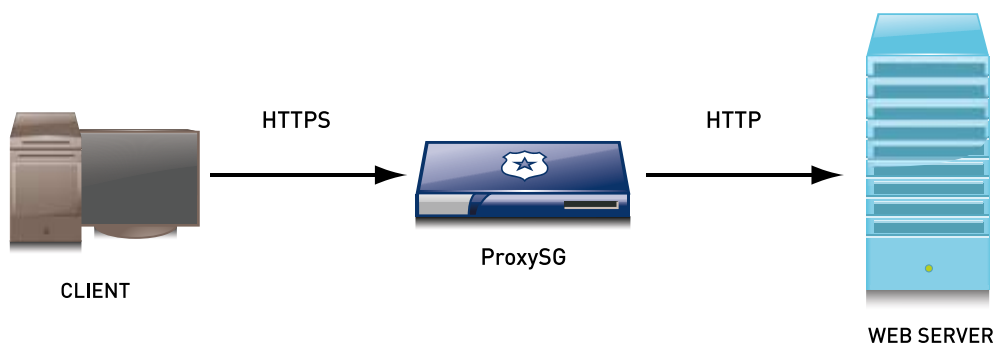
- > The ProxySG terminates the session with the client and establishes another session with the Web server, thereby offloading this process from the Web server
- > The Web server only sees the IP address of the ProxySG; the server identity can be hidden
- > Granular policies with authentication, authorization, and logging can be implemented
- > The ProxySG has built in DOS (Denial of Service) prevention
- > Increased performance with caching provides an improved Web experience

Moreover with SSL proxy configured, the ProxySG terminates the SSL session with the client and forwards traffic to the origin server via HTTP, offloading the Web server of this task. The ProxySG’s flexible advanced forwarding architecture coupled with caching provides organizations a best-of-breed solution to leverage their network infrastructure.

Why Implement Blue Coat Reverse Proxy with SSL?

Reverse SSL proxy on the Blue Coat ProxySG provides flexibility to network administrators in defining scalable secured Web services. HTTPS connections are terminated on the ProxySG.

The ProxySG obtains content not currently in cache from the origin server via HTTP. The following diagram presents this design.



Implementing Reverse Proxy with SSL

There are four tasks to implement Reverse Proxy with SSL on the ProxySG:

- ❶ Configuring the SSL Keyring and Certificate
- ❷ Configuring Advanced Forwarding Hosts
- ❸ Add Web Access to Allow HTTPS Traffic
- ❹ Configuring Advanced Forwarding Rules

NOTE: In the following examples `www.example.com` is used for the example organization's URL (what users enter to reach the corporate website), `www.examplewebserver.com` is used for the web server's hostname and `10.2.3.4` is used for the web server's IP address. As you do the procedures, use your own organization's URL, web server's URL, and web server's IP address.

NOTE: This procedure assumes a DNS-based reverse explicit proxy setup, where the hostname, `www.example.com` in the procedures, resolves to an externally reachable IP address on the reverse proxy.

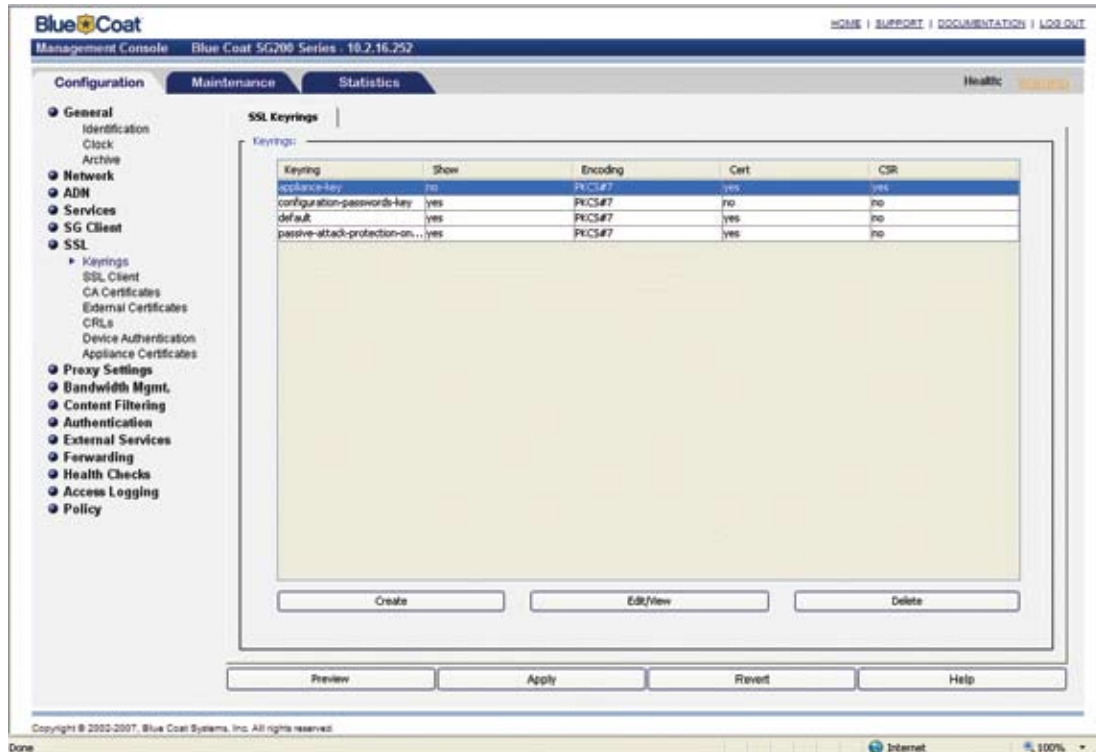
About the Default Proxy Policy

On the Management Console **Configuration > Policy > Policy Options** page you can set the default policy option to **Deny** or **Allow**. The two options provide two different approaches:

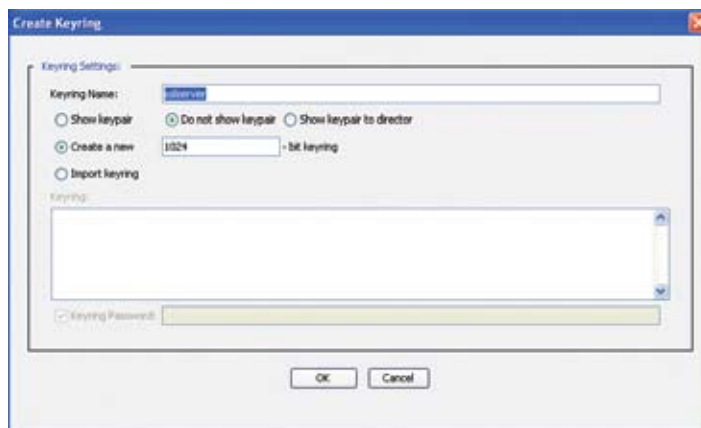
- > A default proxy transaction policy of **Deny** prohibits proxy-type access through the ProxySG appliance; instead, you must create policies to explicitly grant access on a case-by-case basis.
- > A default proxy transaction policy of **Allow** permits most proxy transactions. If your policy is set to **Allow**, you must create policies to explicitly deny access on a case-by-case basis. Please note: if protocol detection is enabled (the default), HTTP CONNECT transactions are only allowed if they are tunneling SSL; if protocol detection is disabled, HTTP CONNECT is only allowed on port 443.

*NOTE: This document assumes the **Deny** default proxy policy so HTTPS traffic must be explicitly configured to be accepted by the proxy. In part three you define policy to allow HTTPS traffic. For more information on developing effective policies, see the [Policy Best Practices](#) tech brief.*

Task 1 – Configuring the SSL Keyring and Certificate

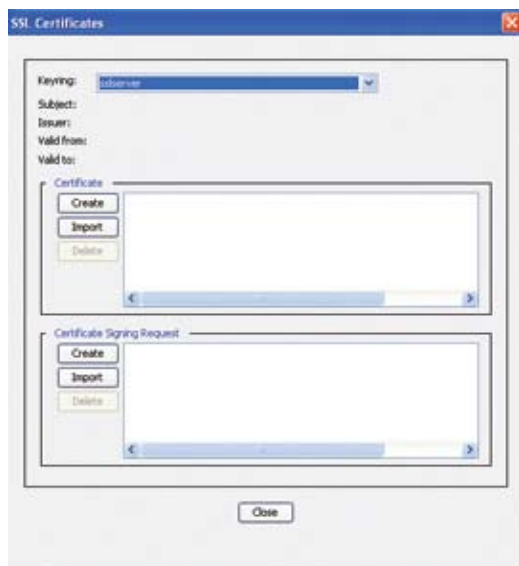


- 1 Open the Blue Coat management console on the ProxySG and go to **SSL > Keyrings** as shown above.



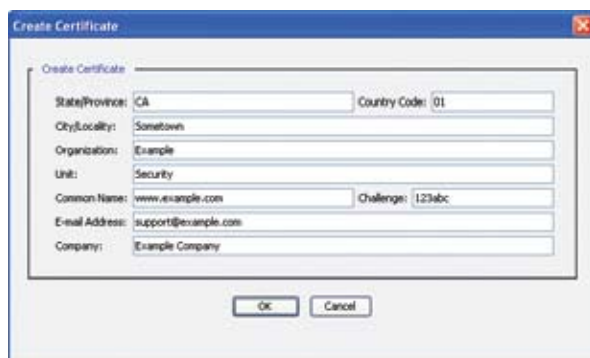
- Click **Create**; the Create Keyring dialog displays. In this example the keyring is named `sslserver`. Click **OK** to create the keyring and dismiss the dialog. Click **Apply** in the management console. Click **OK** to dismiss the confirmation dialog.

Note: If you are going to import a certificate in the next step, then import the keyring for the certificate in this step.



- Next, create or import a certificate for the new keyring. To do this, select the new keyring and click **Edit/View**.

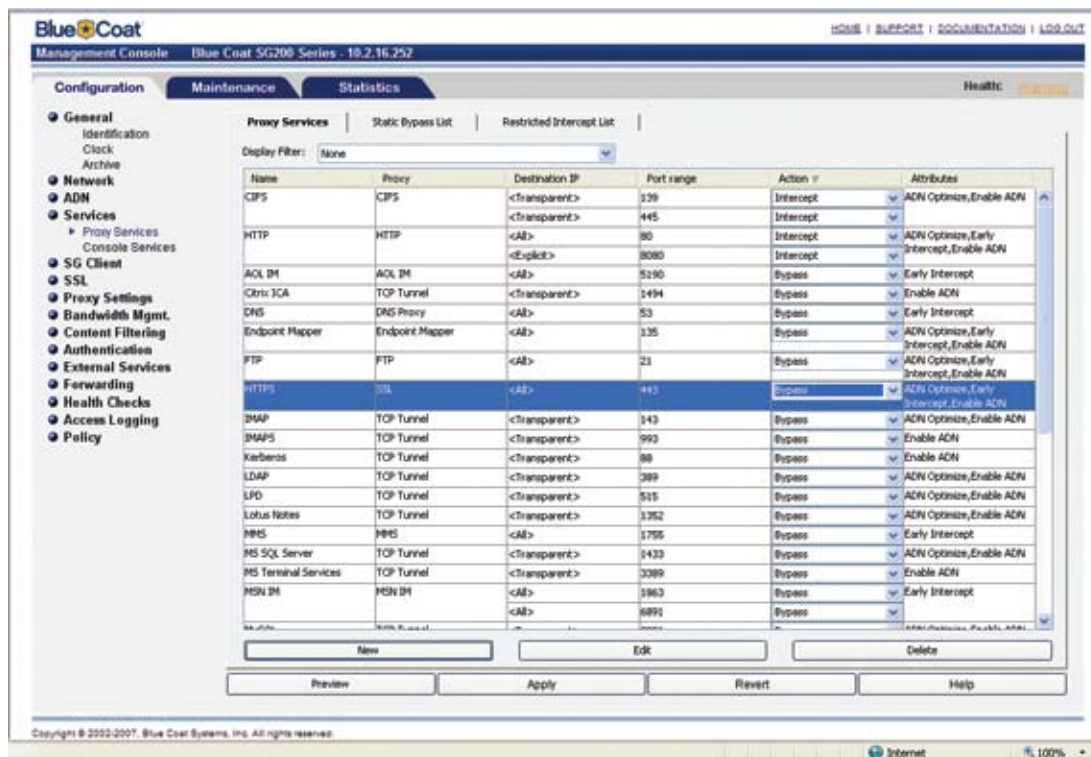
Note: A self-signed certificate can be used (which may cause warnings in the browser) or a request can be made with the Certificate Signing Request option to a valid CA authority that will then return a signed certificate understood by most browsers. This example creates a self-signed certificate.



- Click **Create** in the **Certificate** area and fill in the form. Alternatively, if you want to use a CA certificate and have a signed certificate that you can import into the ProxySG, click **Create** in the **Certificate Singing Request** area instead.

*Note: The **Common Name** is the URL of the web server (hosting the SSL service) that users enter. This example uses `www.example.com`. Click **OK** to finish. The dialog goes away and your data is displayed; if satisfied, click **Close**. Click **Apply** in the management console. Click **OK** to dismiss the confirmation dialog.*

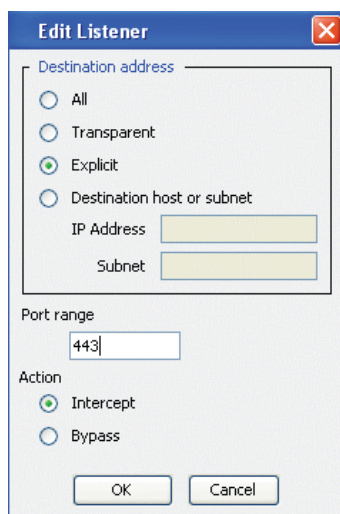
- 5 To view the encrypted key, click **Edit/View**. Click **Close** to dismiss the window.



- 6 Next, the keyring must be assigned to an HTTPS Reverse Proxy service on the ProxySG: From the Blue Coat management console go to **Services > Proxy Services** as shown above.



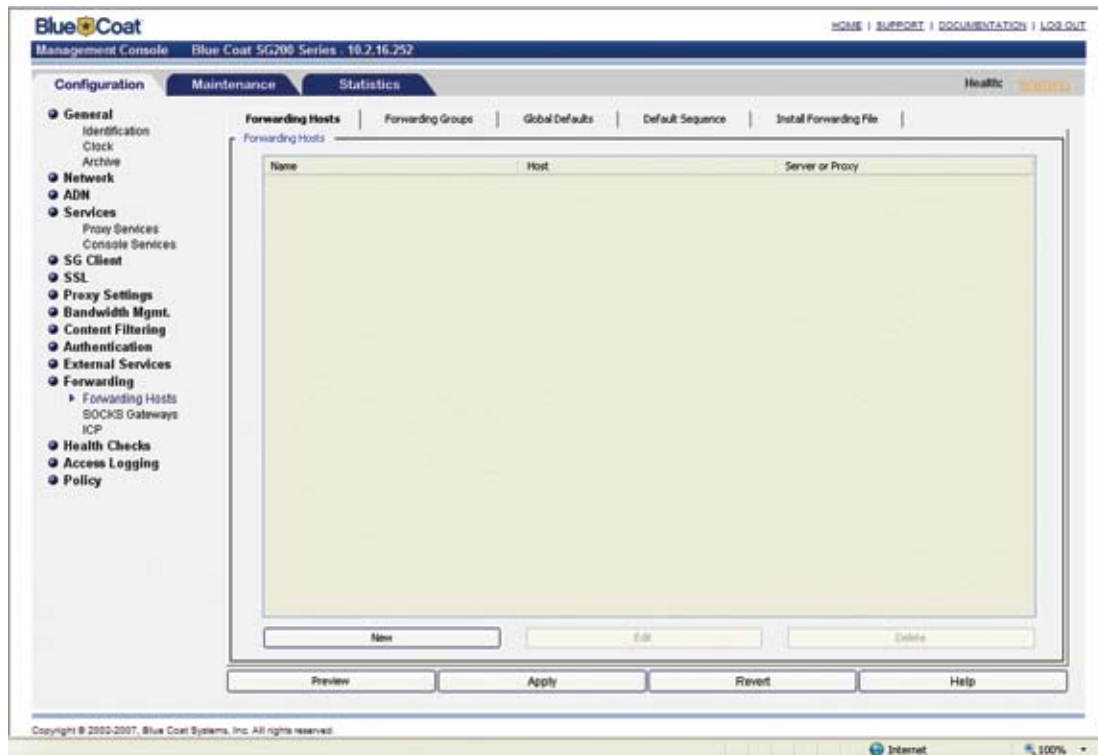
- 7 Select the HTTPS service and click **Edit**. Select HTTPS Reverse Proxy from the **Proxy** dropdown menu; additional options display. Select the previously defined certificate for the **Keyring** option (sslserver in the example).



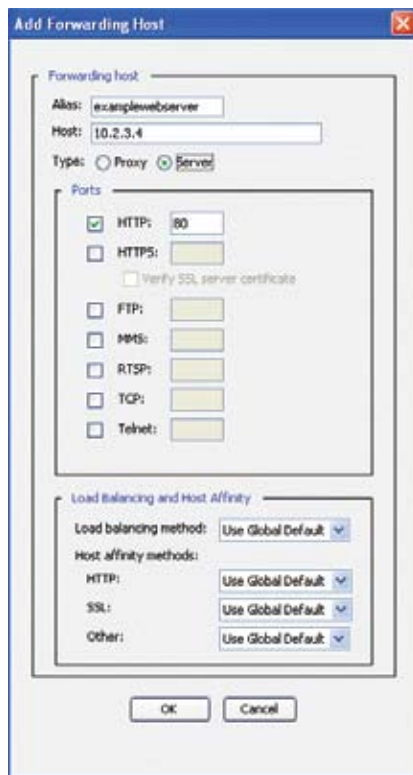
- 8 Ensure that the Listener Destination address is set to **Explicit** and the action is set to **Intercept**. Click **New** to create a new listener if needed, or click **Edit** to modify an existing one. Click **OK** to dismiss the Listener dialog. Click **OK** to dismiss the Edit Service dialog; click **Apply** in the Management Console. Click **OK** to dismiss the confirmation dialog.

Task 2 – Configuring Advanced Forwarding Hosts

The second task in this process is to define the location of the back-end server(s) to obtain the content.

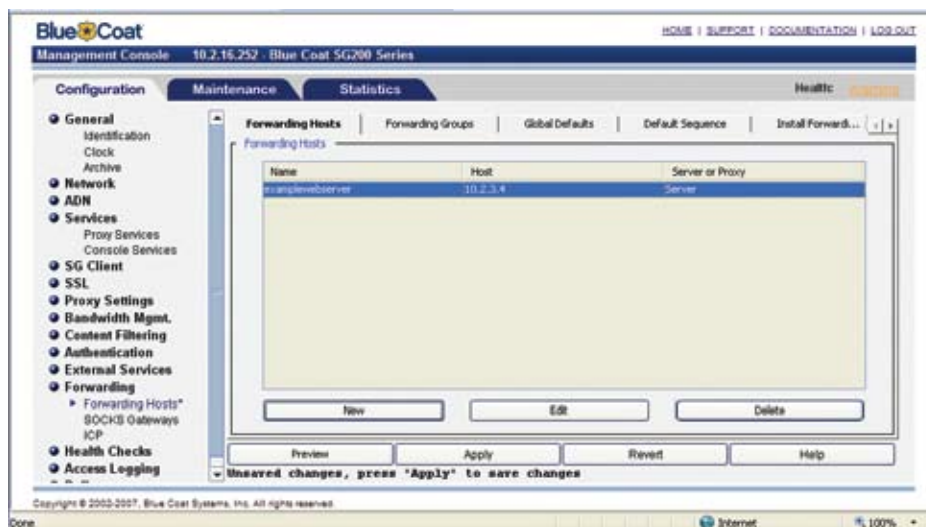


- 1 From the management console, go to **Forwarding > Forwarding Hosts** as shown above.



The "Add Forwarding Host" dialog box is shown. It has a title bar with a close button. The "Forwarding host" section contains fields for "Alias" (examplewebserver) and "Host" (10.2.3.4). The "Type" section has radio buttons for "Proxy" and "Server", with "Server" selected. The "Ports" section has checkboxes for "HTTP" (checked), "HTTPS", "FTP", "MMS", "RTSP", "TCP", and "Telnet", each with a corresponding port number field. Below the ports is a checkbox for "Verify SSL server certificate". The "Load Balancing and Host Affinity" section has dropdown menus for "Load balancing method", "HTTP", "SSL", and "Other", all set to "Use Global Default". At the bottom are "OK" and "Cancel" buttons.

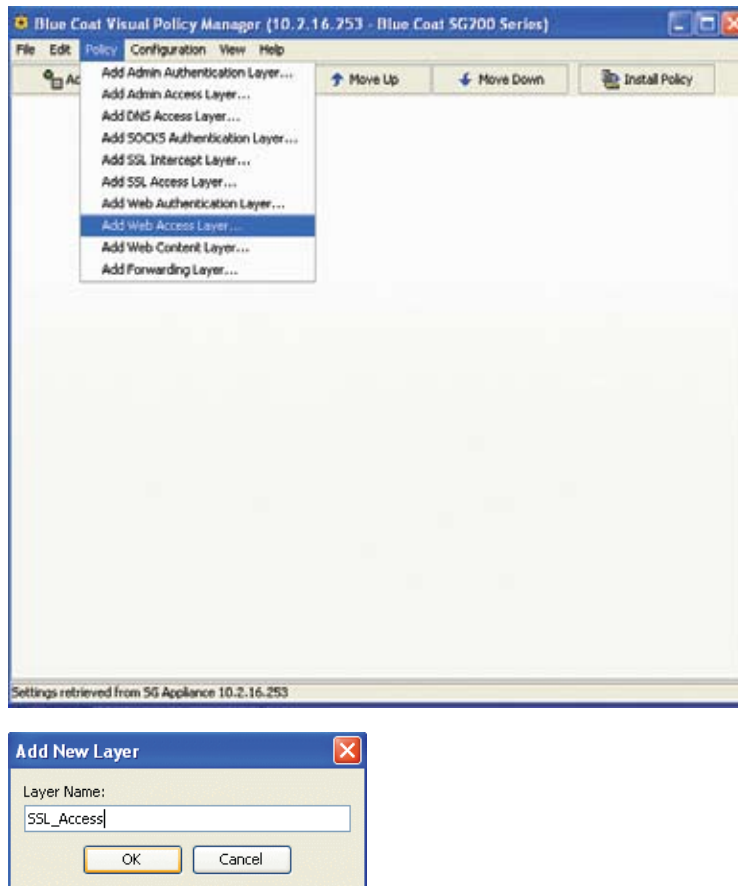
- 2 Click **New**. The Add Forwarding Host dialog displays. Enter the name of your web server as the **Alias** and the IP address of your web server as the **Host**. Select **Server** as the **Type** option and click **OK**.



- 3 Click **Apply** to finish and **OK** to dismiss the confirmation box.

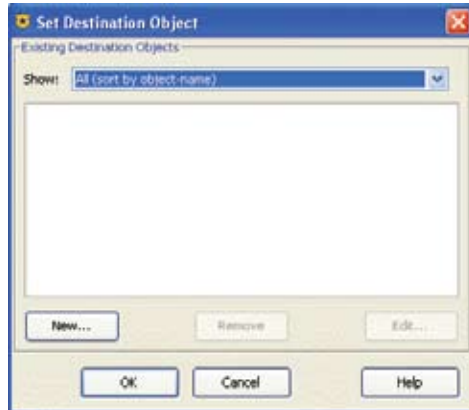
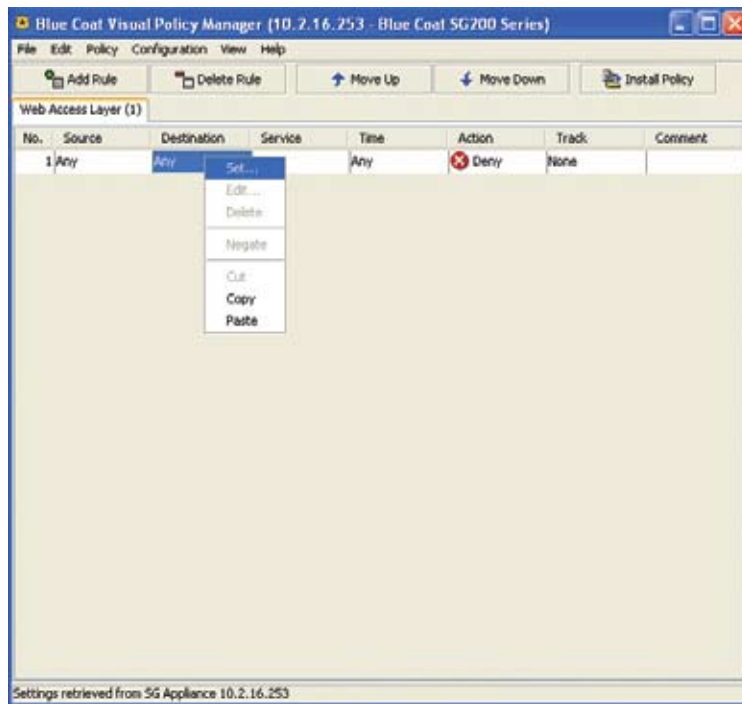
Task 3 – Add Web Access to Allow HTTPS Traffic

Launch the Visual Policy Manager (VPM) to create a Web Access Layer that allows HTTPS traffic to the web server.

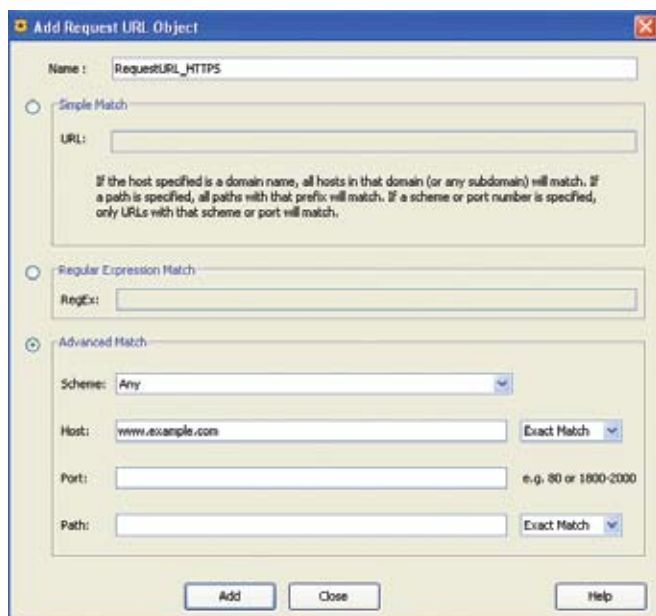


- 1 Begin by clicking **Policy** and selecting **Add Web Access Layer** from the drop-down list. Name the layer, `SSL_Access`; for example, and click **OK**. The layer displays in the VPM.

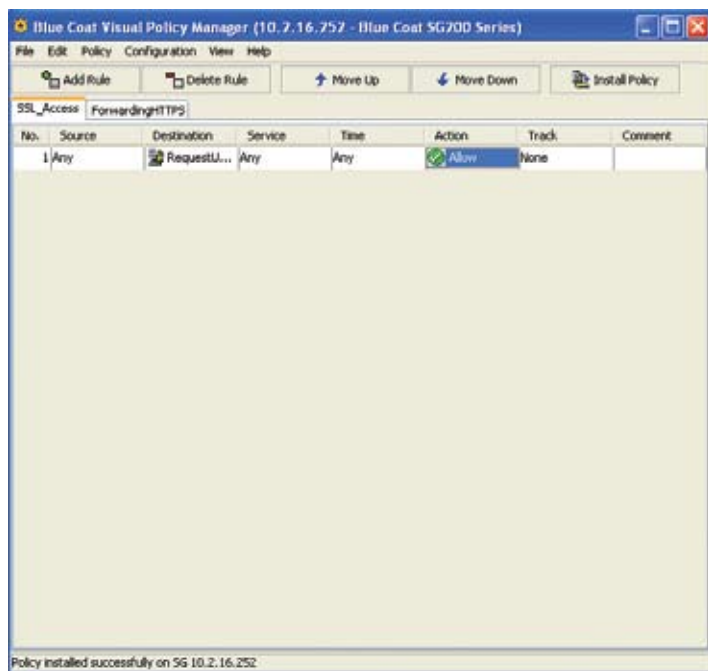
Note: To help maintain scalability, Blue Coat recommends giving relevant names to layers and objects.



- 2 Right-click the **Destination** setting and select **Set**. The Set Destination dialog displays.



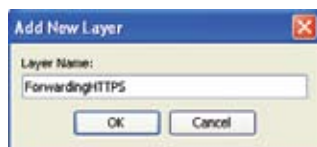
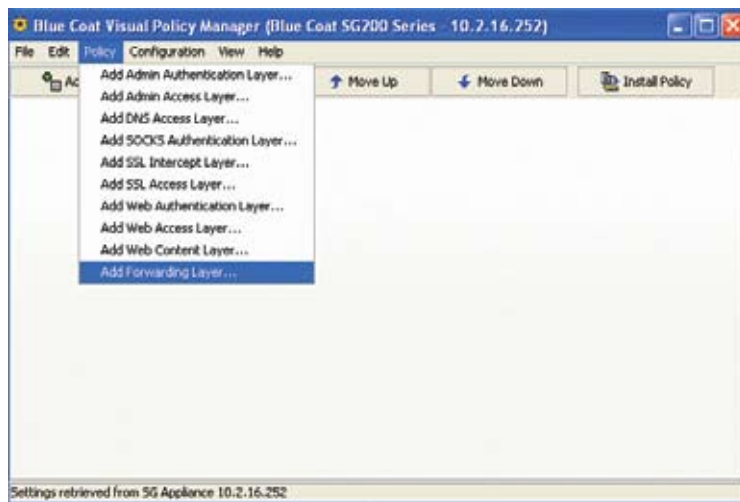
- 3 Click **New** and select **Request URL**. The Add Request URL Object dialog displays. Click **Advanced Match** and select **Any** for **Scheme**. Enter your organization's URL for the **Host**. Name the object; RequestURL_Https for example. Click **Add** to add the object (wait until the dialog clears); click **Close** to dismiss the dialog. The Set Destination Object dialog re-displays with the new object.
- 4 Click **OK** to set the object and dismiss the dialog. The VPM re-displays with the new **Destination** setting.



- 5 Next, right-click the **Action** setting and select **Allow**. Click **Install Policy** to install the Web Access Layer to the policy. Click **OK** to dismiss the confirmation dialog.
- Keep the VPM open for the next procedure.

Task 4 – Configuring Advanced Forwarding Rules

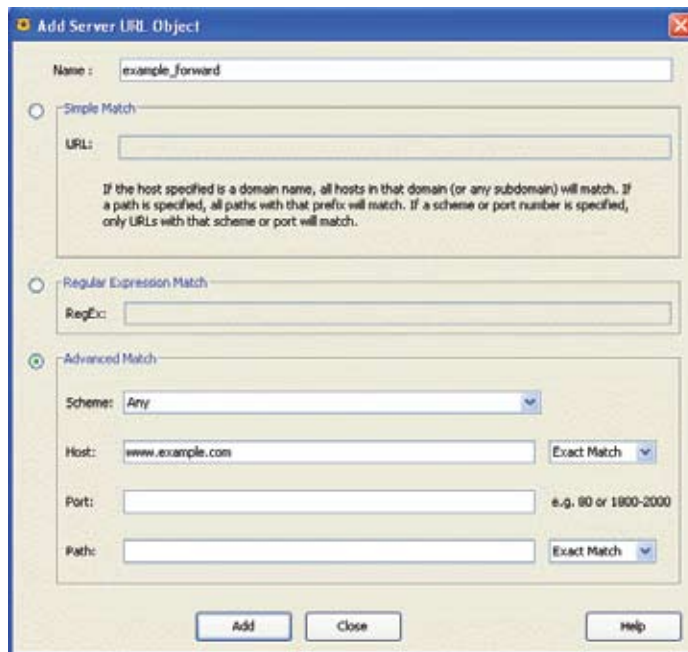
Next, add the forwarding rules to send requests to the web server.



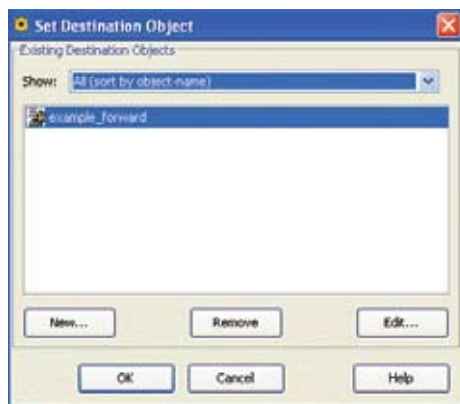
- 1 Begin by clicking **Policy** and selecting **Add Forwarding Layer** from the dropdown menu. Name the layer ForwardingHTTPS; for example. Click **OK** to add the new layer. The new layer displays in the VPM.
- 2 Right-click the **Destination** setting and select **Set**. The Set Destination Object dialog displays.



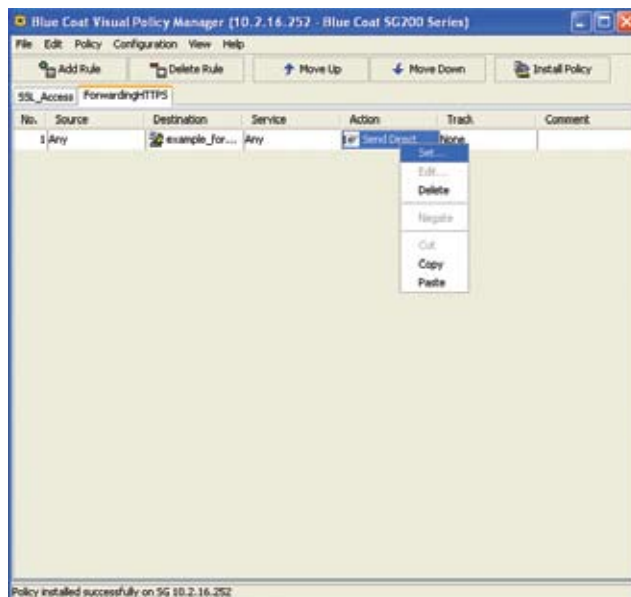
- 3 Click **New** and select **Server URL**. The Add Server URL Object dialog displays.



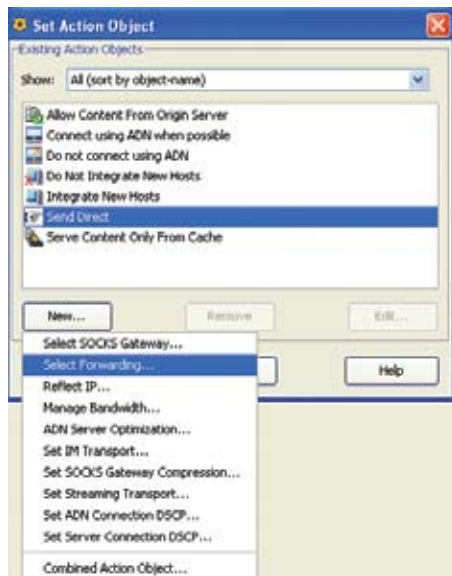
- 4 Click **Advanced Match**, select **Any** as the **Scheme** and enter your organization's URL as the **Host**. Name the object; `example_forward`; for example. Click **Add** to add the Server URL object (wait until the dialog clears). Click **Close** to dismiss the dialog. The Set Destination Object dialog re-displays.



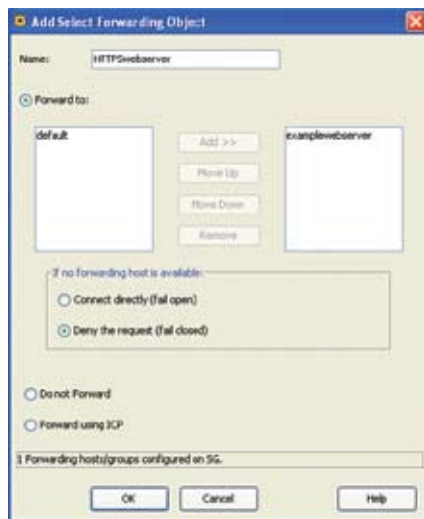
- 5 Click **OK** to set the object and dismiss the dialog. The new destination object displays in the forwarding layer.



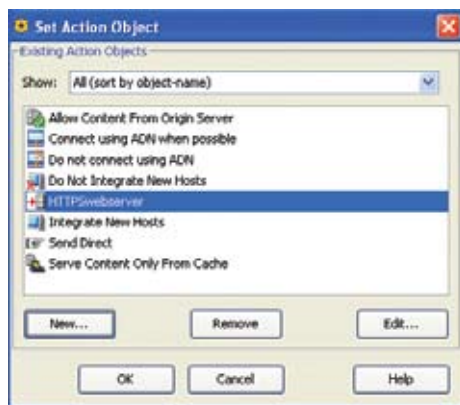
- 6 Next, define forwarding of HTTPS requests to your web server: Right-click the **Action** setting and select **Set**. The Set Action Object dialog displays.



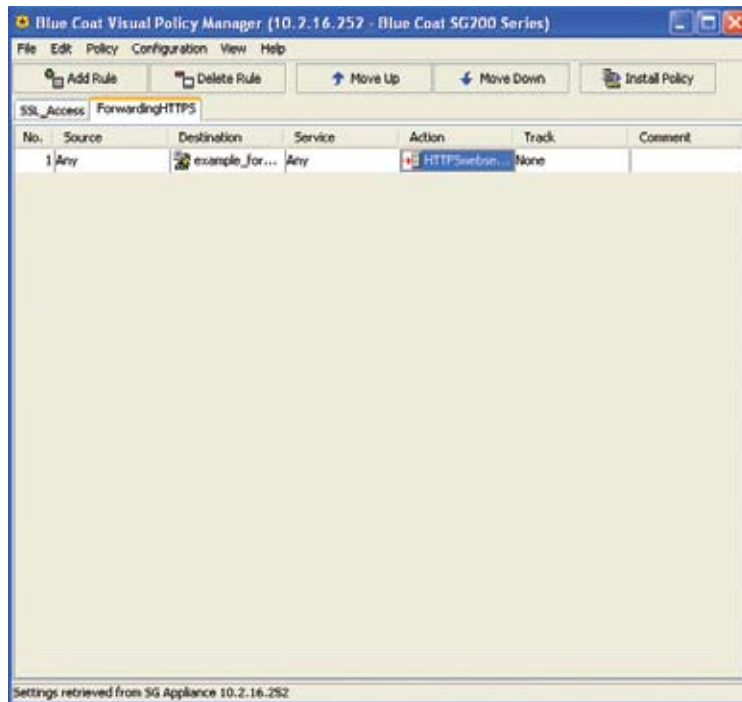
- 7 Click **New** and select **Select Forwarding**. The Add Select Forwarding Object dialog displays.



- 8 Select your previously-defined forwarding host (examplewebserver, in the example) and click **Add** to move it from the left-hand box to the right-hand box. Name the forwarding object HTTPSwbserver; for example. Click **OK** to add the object and dismiss the dialog. The Set Action Object dialog re-displays.



- 9 Click **OK** to set the object and dismiss the dialog. The VPM displays the configured forwarding layer as shown below.



- ⑩ Click **Install Policy** to install the forwarding policy. Click **OK** to dismiss the confirmation dialog. Close the VPM window.

In this example, now users requesting `https://www.example.com` receive content from `http://www.examplewebserver.com`.

Note: Make sure that `www.example.com` resolves to the IP address of your ProxySG.

Testing Your Configuration

The last step is to validate that reverse proxy with SSL is working. Open a browser and go to `https://www.example.com`. You see the secured content of `www.examplewebserver.com`.



Caveats

It is possible that the origin Web server references links to HTTP instead of relative links. This may cause certain pages to reference the origin Web server. In this case you need to install the following local policy (use the **Policy > Policy Files** page) that uses the `TwoWayURLRewrite` function. Be sure to change the variables appropriately.

```
<Proxy>
url.host=www.example.com action.bluecoat_server_portal(yes)

; This transformation provides server response rewriting.
; Note that in addition to rewrite_url_prefix,
; rewrite_url_substring and rewrite_script_substring are available.
define url_rewrite bluecoat_portal
rewrite_url_prefix "https://www.example.com/"
"http://www.examplewebserver.com/"
end

; This action runs the transform for the web server portaling for http
; content
; Note that the action is responsible for rewriting related headers
define action web_server_portal
; request rewriting
rewrite( url, "^https://www\\.example\\.com/(.*)",
"http://www.examplewebserver.com/${1}", cache, server )
rewrite( request.header.Referer, "^https://www\\.example\\.com/(.*)",
"http://www.examplewebserver.com/${1}" )
; response rewriting
transform examplewebserver_portal
end
```

Note: The server_url you defined in the VPM forwarding layer previously will need to reflect the name after the rewrite; www.examplewebserver.com, in this example.

Conclusion

The ProxySG provides powerful reverse proxy capabilities allowing an organization to terminate SSL on the ProxySG and make content requests in behalf of the user to the origin server. An enterprise can achieve greater performance and SSL security benefits by implementing a reverse proxy with SSL solution on their network.