Configuring Load Balancing for a Service

https://campus.barracuda.com/doc/4259865/

A load balancing policy allows the service to distribute the traffic to an appropriate server based on the configured load balancing algorithm. To configure the load balancing policy for a service, navigate to the **BASIC > Services** page. Identify the service in the **Services** list that you want to configure to load balance, and click **Edit** next to it. Specify values for the following fields in the **Load Balance** section:

- **Algorithm** – Select the algorithm to be used to distribute incoming requests for the service.
  - **Round Robin** – Incoming requests are equally distributed among all servers.
  - **Weighted Round Robin** – Requests are distributed in proportion to server weights which you define when you add a server. A server with a higher weight will be sent proportionally more requests. For information on configuring a server, see How to Add a Real Server .
  - **Least Requests**– Load balances the incoming requests based on the number of outstanding requests to the servers. The requests are distributed to the server with the fewest outstanding requests.
- **Persistence Method** – Select the persistence method to be used to maintain the connection between a client and the first server that it connects to, even for load balanced traffic. Persistence is required when the server needs to maintain state information about every client (for example, login information).
  - **None** – No persistence configured. All requests, regardless of whether they originate from the same client or a different one, are load balanced according to the algorithm.
  - **Source IP** – Based on the source IP address of the client, the request is routed to a server. The source IP mask defines how to group the client IP addresses. The first request from a group is load balanced and routed to a server. Subsequent requests from the same group are routed to the same server to which the first request was routed.
  - **Cookie Insert** – The Barracuda Web Application Firewall routes the first request from a client to one of the servers based on the load balancing algorithm. At the same time, it inserts a cookie to identify the client. Subsequent requests from the client include the persistence cookie, so they can be routed to the same server as the first request was.
  - **Cookie Passive** – Similar to Cookie Insert, except that only the server inserts the cookie if needed. This provides additional optimization because requests are load balanced normally unless there is a requirement to persist a session, which is indicated by the presence of a cookie.
  - **HTTP Header** – All incoming HTTP requests are directed to the same Real Server based on the value of the header specified in the Header Name field. The application (e.g., Microsoft Exchange) specifies the header name. Header Name is case-sensitive.
  - **URL Parameter** – All incoming HTTP requests are directed to the same Real Server

based on the value of the parameter specified in the Parameter Name field. Parameter Name is case-sensitive.

- **Failover Method**– Select the method to handle persistent client requests when the server that handled the original request is out-of-service.
  - **Load Balance** – The requests are load balanced between the "alive" servers.
  - **Error**– Sends out "503 service unavailable" error message. This method is not supported for the persistence method **Source IP**.
- **Source IP Netmask** – Enter the netmask for the **Source IP** persistence method. The IP plus netmask results in a network identifier that is used to identify a client. A more specific netmask (such as 255.255.255.255) will track each client independently, and may cause a higher memory load on the Barracuda Web Application Firewall. On the other hand, a less specific netmask (such as 255.255.0.0) will group multiple clients under the same network identifier and connect them all to the same server.
- **Persistence Cookie Name** – The name of the cookie that will be used for persistence. This is used when **Persistence Method** is set to **Cookie Insert**.
- **Persistence Cookie Path** – Enter the path property of the persistency cookie.
- **Persistence Cookie Domain** – Enter the domain name of the server of a persistency cookie.
- **Cookie Age** – Specify the expiry age of the persistence cookie in minutes. This ensures sessions are no longer persisted after this interval when **Cookie Passive** or **Cookie Insert** is chosen as the Persistence Method.
- **Persistence Idle Timeout** – Sets the maximum idle time (in seconds) for a persistent connection. A client is directed to the same Real Server unless the connection is inactive for more than the specified number of seconds. Example: Consider a service is configured with two Real Servers, and **Persistence Idle Timeout** is set to 1200 seconds (20 minutes). When a client sends a request to this service, it is directed to the first available Real Server based on the configured load balancing algorithm. All subsequent requests are forwarded to the same Real Server. If no request is sent for a time exceeding **Persistence Idle Timeout**, then new requests from the client may be forwarded to either Real Server.
- **Header Name** – The name of the header for which the value needs to be checked in the HTTP requests.
- **Parameter Name** – The name of the parameter for which the value needs to be checked in the URL.

If a content rule is present under the service, the load balancing settings under the content rule will take precedence over the service level load balancing settings.

Persisting Traffic across HTTP and HTTPS Services

You can maintain persistence across HTTP and HTTPS services on the Barracuda Web Application

Firewall using the **Cookie Insert** persistence method.

If you have two services for the same web application, where one listens on port 80 (HTTP) and the other on port 443 (HTTPS), you can persist the traffic to the same server when the client sends a request to a HTTP service, and the subsequent request sent to a HTTPS service in the web application. This can be achieved by performing the configurations below:

1. Create two (2) services, a HTTP and a HTTPS service in the **BASIC > Services** page. Ensure both the services are configured with the same sub set of server(s).
2. Edit both the services and do the following:
    1. Set **Persistence Method** to *Cookie Insert*.
    2. Configure **Persistence Cookie Name** and **Persistence Cookie Path**. Ensure that both the services have same cookie name and cookie path.
3. Set **Persistence Across Services** to *Yes* in the **ADVANCED > System Configuration** page, **Advanced** section.

Use Case

Consider *www.example.com* is an e-commerce website in which product viewing and shopping are in the HTTP space, and purchase/checkout is in the HTTPS space. Initially when a client access *www.example.com*, the request is landed in the product viewing page *http://www.example.com/us/en_US/products* . After selecting the products, the client navigates to the shopping cart to check the list, *http://www.example.com/us/en_US/cart* . Here, the product viewing and shopping cart are in the HTTP space. The client moves to the purchase/checkout page *https://www.example.com/us/en_US/login/checkout* to buy the selected products. The checkout page is in the HTTPS space.

To ensure persisting traffic between HTTP and HTTPS services, you must configure persistence on the Barracuda Web Application Firewall for the web application.

Here, *http://example.com* is hosted on service 1 (port 80), and *https://example.com* is hosted on service 2 (port 443).

In the above scenario, when the client selects the cart, the request is served by service 1/server 1. When the client accesses the checkout page, the request should be served by service 2/server 1. To achieve this, persistence should be configured between the two services on the Barracuda Web Application Firewall.

The cookie name and cookie path for both the services (service 1 and service 2) are configured as:

- **Cookie Name**: persistence
- **Cookie Path**: example.com

Persistence across HTTP and HTTPS services is possible only when **Persistence Across Services** is enabled on the Barracuda Web Application Firewall. Therefore, ensure that you set **Persistence Across Services** to *Yes* in the **ADVANCED > System Configuration** page, **Advanced** section.

If one of the server of service 1 is not part of service 2, then the request can be served by any server depending on the load balancing algorithm.