**Sticky Sessions with Classic LB**

By default, a Classic Load Balancer routes each request independently to the registered instance with the smallest load. However, you can use the *sticky session* feature (also known as *session affinity*), which enables the load balancer to bind a user's session to a specific instance. This ensures that all requests from the user during the session are sent to the same instance.

The key to managing sticky sessions is to determine how long your load balancer should consistently route the user's request to the same instance. If your application has its own session cookie, then you can configure Elastic Load Balancing so that the session cookie follows the duration specified by the application's session cookie. If your application does not have its own session cookie, then you can configure Elastic Load Balancing to create a session cookie by specifying your own stickiness duration.

Elastic Load Balancing creates a cookie, named AWSELB, that is used to map the session to the instance.

**Requirements**

* An HTTP/HTTPS load balancer.
* At least one healthy instance in each Availability Zone.

**Compatibility**

* The RFC for the path property of a cookie allows underscores. However, Elastic Load Balancing URI encodes underscore characters as %5F because some browsers, such as Internet Explorer 7, expect underscores to be URI encoded as %5F. Because of the potential to impact browsers that are currently working, Elastic Load Balancing continues to URI encode underscore characters. For example, if the cookie has the property path=/my\_path, Elastic Load Balancing changes this property in the forwarded request to path=/my%5Fpath.
* You can't set the secure flag or HttpOnly flag on your duration-based session stickiness cookies. However, these cookies contain no sensitive data. Note that if you set the secure flag or HttpOnly flag on an application-controlled session stickiness cookie, it is also set on the AWSELB cookie.
* If you have a trailing semicolon in the Set-Cookie field of an application cookie, the load balancer ignores the cookie.

**Contents**

* [Duration-Based Session Stickiness](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-sticky-sessions.html#enable-sticky-sessions-duration)
* [Application-Controlled Session Stickiness](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-sticky-sessions.html#enable-sticky-sessions-application)

**Duration-Based Session Stickiness**

The load balancer uses a special cookie to track the instance for each request to each listener. When the load balancer receives a request, it first checks to see if this cookie is present in the request. If so, the request is sent to the instance specified in the cookie. If there is no cookie, the load balancer chooses an instance based on the existing load balancing algorithm. A cookie is inserted into the response for binding subsequent requests from the same user to that instance. The stickiness policy configuration defines a cookie expiration, which establishes the duration of validity for each cookie. The load balancer does not refresh the expiry time of the cookie and does not check whether the cookie is expired before using it. After a cookie expires, the session is no longer sticky. The client should remove the cookie from its cookie store upon expiry.

If an instance fails or becomes unhealthy, the load balancer stops routing requests to that instance, and chooses a new healthy instance based on the existing load balancing algorithm. The request is routed to the new instance as if there is no cookie and the session is no longer sticky.

If a client switches to a listener with a different backend port, stickiness is lost.

**To enable duration-based sticky sessions for a load balancer using the console**

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. On the navigation pane, under **LOAD BALANCING**, choose **Load Balancers**.
3. Select your load balancer.
4. On the **Description** tab, choose **Edit stickiness**.
5. On the **Edit stickiness**page, select **Enable load balancer generated cookie stickiness**.
6. (Optional) For **Expiration Period**, type the cookie expiration period, in seconds. If you do not specify an expiration period, the sticky session lasts for the duration of the browser session.
7. Choose **Save**.

**To enable duration-based sticky sessions for a load balancer using the AWS CLI**

1. Use the following [create-lb-cookie-stickiness-policy](http://docs.aws.amazon.com/cli/latest/reference/elb/create-lb-cookie-stickiness-policy.html) command to create a load balancer-generated cookie stickiness policy with a cookie expiration period of 60 seconds:

**aws elb create-lb-cookie-stickiness-policy --load-balancer-name *my-loadbalancer* --policy-name *my-duration-cookie-policy* --cookie-expiration-period 60**

1. Use the following [set-load-balancer-policies-of-listener](http://docs.aws.amazon.com/cli/latest/reference/elb/set-load-balancer-policies-of-listener.html) command to enable session stickiness for the specified load balancer:

**aws elb set-load-balancer-policies-of-listener --load-balancer-name *my-loadbalancer* --load-balancer-port *443* --policy-names *my-duration-cookie-policy***

**Note**

The set-load-balancer-policies-of-listener command replaces the current set of policies associated with the specified load balancer port. Every time you use this command, specify the --policy-names option to list all policies to enable.

1. (Optional) Use the following [describe-load-balancers](http://docs.aws.amazon.com/cli/latest/reference/elb/describe-load-balancers.html) command to verify that the policy is enabled:

**aws elb describe-load-balancers --load-balancer-name *my-loadbalancer***

The response includes the following information, which shows that the policy is enabled for the listener on the specified port:

{

"LoadBalancerDescriptions": [

{

...

"ListenerDescriptions": [

{

"Listener": {

"InstancePort": 443,

"SSLCertificateId": "arn:aws:iam::123456789012:server-certificate/my-server-certificate",

"LoadBalancerPort": 443,

"Protocol": "HTTPS",

"InstanceProtocol": "HTTPS"

},

"PolicyNames": [

"my-duration-cookie-policy",

"ELBSecurityPolicy-2016-08"

]

},

...

],

...

"Policies": {

"LBCookieStickinessPolicies": [

{

"PolicyName": "my-duration-cookie-policy",

"CookieExpirationPeriod": 60

}

],

"AppCookieStickinessPolicies": [],

"OtherPolicies": [

"ELBSecurityPolicy-2016-08"

]

},

...

}

]

}

**Application-Controlled Session Stickiness**

The load balancer uses a special cookie to associate the session with the instance that handled the initial request, but follows the lifetime of the application cookie specified in the policy configuration. The load balancer only inserts a new stickiness cookie if the application response includes a new application cookie. The load balancer stickiness cookie does not update with each request. If the application cookie is explicitly removed or expires, the session stops being sticky until a new application cookie is issued.

If an instance fails or becomes unhealthy, the load balancer stops routing requests to that instance, and chooses a new healthy instance based on the existing load balancing algorithm. The load balancer treats the session as now "stuck" to the new healthy instance, and continues routing requests to that instance even if the failed instance comes back.

**To enable application-controlled session stickiness using the console**

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. On the navigation pane, under **LOAD BALANCING**, choose **Load Balancers**.
3. Select your load balancer.
4. On the **Description** tab, choose **Edit stickiness**.
5. On the **Edit stickiness** page, select **Enable application generated cookie stickiness**.
6. For **Cookie Name**, type the name of your application cookie.
7. Choose **Save**.

**To enable application-controlled session stickiness using the AWS CLI**

1. Use the following [create-app-cookie-stickiness-policy](http://docs.aws.amazon.com/cli/latest/reference/elb/create-app-cookie-stickiness-policy.html) command to create an application-generated cookie stickiness policy:

**aws elb create-app-cookie-stickiness-policy --load-balancer-name *my-loadbalancer* --policy-name *my-app-cookie-policy* --cookie-name *my-app-cookie***

1. Use the following [set-load-balancer-policies-of-listener](http://docs.aws.amazon.com/cli/latest/reference/elb/set-load-balancer-policies-of-listener.html) command to enable session stickiness for a load balancer:

**aws elb set-load-balancer-policies-of-listener --load-balancer-name *my-loadbalancer* --load-balancer-port *443* --policy-names *my-app-cookie-policy***

**Note**

The set-load-balancer-policies-of-listener command replaces the current set of policies associated with the specified load balancer port. Every time you use this command, specify the --policy-names option to list all policies to enable.

1. (Optional) Use the following [describe-load-balancers](http://docs.aws.amazon.com/cli/latest/reference/elb/describe-load-balancers.html) command to verify that the sticky policy is enabled:

**aws elb describe-load-balancers --load-balancer-name *my-loadbalancer***

1. The response includes the following information, which shows that the policy is enabled for the listener on the specified port:
2. {
3. "LoadBalancerDescriptions": [
4. {
5. ...
6. "ListenerDescriptions": [
7. {
8. "Listener": {
9. "InstancePort": 443,
10. "SSLCertificateId": "arn:aws:iam::123456789012:server-certificate/my-server-certificate",
11. "LoadBalancerPort": 443,
12. "Protocol": "HTTPS",
13. "InstanceProtocol": "HTTPS"
14. },
15. "PolicyNames": [
16. "my-app-cookie-policy",
17. "ELBSecurityPolicy-2016-08"
18. ]
19. },
20. {
21. "Listener": {
22. "InstancePort": 80,
23. "LoadBalancerPort": 80,
24. "Protocol": "TCP",
25. "InstanceProtocol": "TCP"
26. },
27. "PolicyNames": []
28. }
29. ],
30. ...
31. "Policies": {
32. "LBCookieStickinessPolicies": [],
33. "AppCookieStickinessPolicies": [
34. {
35. "PolicyName": "my-app-cookie-policy",
36. "CookieName": "my-app-cookie"
37. }
38. ],
39. "OtherPolicies": [
40. "ELBSecurityPolicy-2016-08"
41. ]
42. },
43. ...
44. }
45. ]

}