

Cookie switching and HTTP header insertion together with SSL offload

Summary

Header insertion to distinguish whether the incoming connection was received on HTTP or SSL together with cookie switching/insertion for persistency reasons (to bind clients to the same backend server for HTTP and SSL traffic).

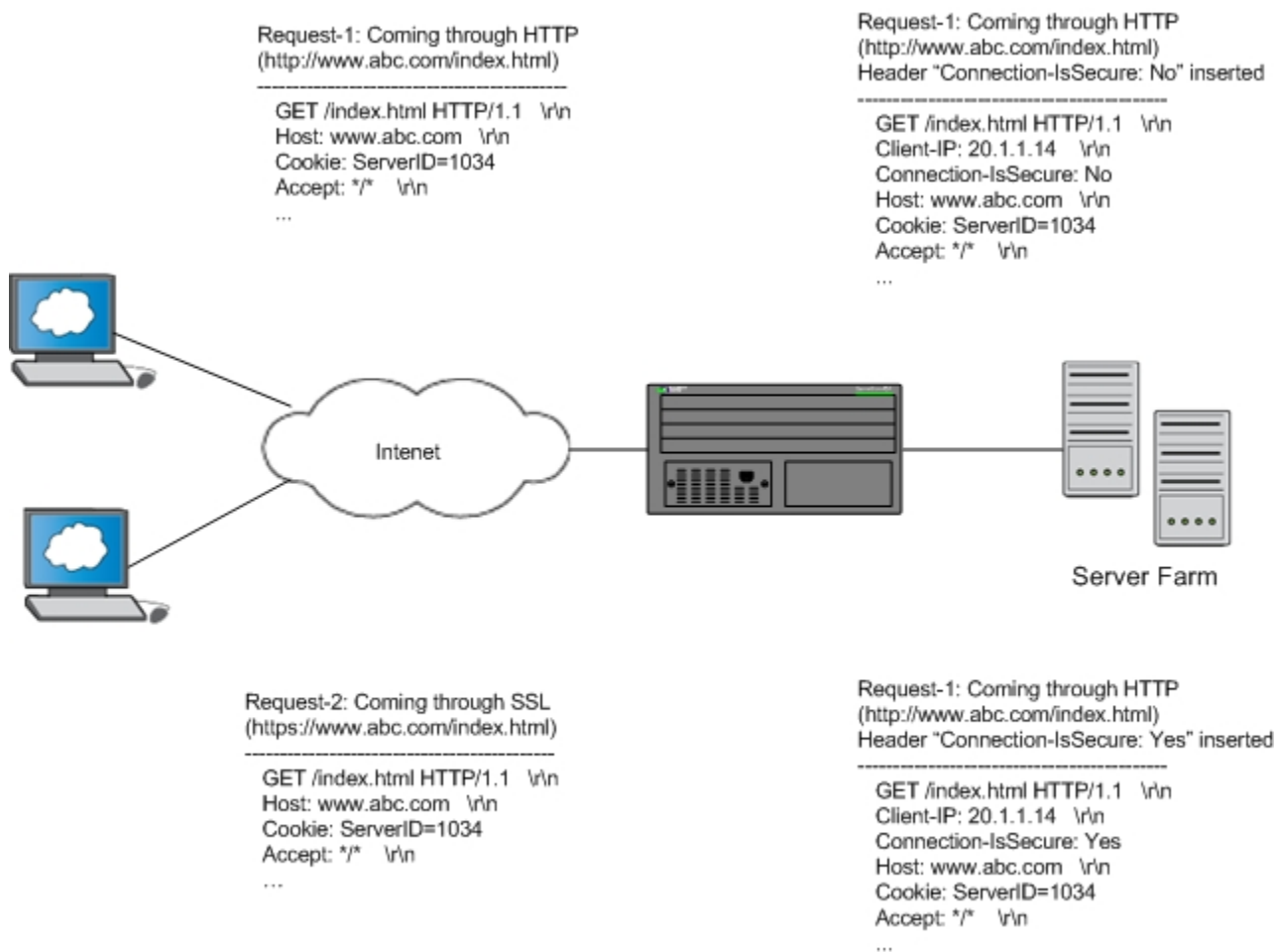
Specifics

There are a lot of setups which are using multiple front-end ports like HTTP and SSL which are bound to a single backend port like the HTTP port. A special header may be inserted in all requests being forwarded to the real-server to specify whether the connection was received on SSL, or on HTTP.

We are going to use a header with the name "Connection-IsSecure" and the values "Yes" or "No".

The example below is using cookie persistency on top of HTTP header insertion. Cookie persistency is used to gain persistency for the HTTP and SSL services and as well for clients moving from HTTP to SSL.

Topology



Sample Script/Code/Configuration

ssl profile verisign128

keypair-file verisign128key

certificate-file verisign128cert

cipher-suite all-cipher-suites

Cookie switching and HTTP header insertion together with SSL offload

session-cache off

!

csw-rule "cookie-persist" header "Cookie" search "ServerID="

!

csw-policy "policy-http"

match "cookie-persist" persist offset 0 length 4 group-or-server-id

match "cookie-persist" rewrite request-insert header "Connection-IsSecure: NO"

match "cookie-persist" rewrite request-insert client-ip

default forward 1

default rewrite insert-cookie "ServerID"

default rewrite request-insert header

default rewrite request-insert client-ip

csw-policy "policy-ssl"

match "cookie-persist" persist offset 0 length 4 group-or-server-id

match "cookie-persist" rewrite request-insert header "Connection-IsSecure: YES"

match "cookie-persist" rewrite request-insert client-ip

default forward 1

default rewrite insert-cookie "ServerID"

default rewrite request-insert header

default rewrite request-insert client-ip

!

Cookie switching and HTTP header insertion together with SSL offload

server real rs18 20.1.1.18

port http

port http url "HEAD /"

port http server-id 1024

port http group-id 1 1

port 8080

port 8080 server-id 1024

port 8080 group-id 1 1

!

server real rs19 20.1.1.19

port http

port http url "HEAD /"

port http server-id 1025

port http group-id 1 1

port ssl

port 8080

port 8080 server-id 1025

port 8080 group-id 1 1

!

server virtual vip20 20.1.1.10

port http

port http csw-policy "policy-http"

Cookie switching and HTTP header insertion together with SSL offload

```
port http csw
```

```
port ssl
```

```
no port ssl sticky
```

```
port ssl ssl-terminate verisign128
```

```
port ssl csw-policy "policy-ssl"
```

```
port ssl csw
```

```
bind http rs18 http rs19 http
```

```
bind ssl rs18 8080 real-port http rs19 8080 real-port http
```

ATTENTION: This requires SSL offload - ensure you are using at least ADX OS >= 12.1.

Verification

- show server bind

- show csw-policy policy1

Verify that the real servers are bound to the virtual server and verify that they are UP (Active):

```
SLB-ServerIron#show server binding Bind info Virtual server: vip20
```

```
Status: enabled IP: 20.1.1.10 http -----> rs18: 20.
```

Cookie switching and HTTP header insertion together with SSL offload

"show csw-policy" shows detailed information about policy and rules, the times they were hit etc. In our case there were 9 hits for the persistency part of the policy and a single hit for the default path:

```
SLB-ServerIron#sho csw-policy policy-ssl Policy Name      : policy-ssl Policy Type      : Content Switching Pol
SLB-ServerIron#
```