Authenticating Client Using HTTP Cookie

Use Case:

Most of the applications today generate HTTP cookie post authentication. If authentication is successful, a cookie is generated and sent to client. On subsequent requests from same client, if the AUTH cookie is present and valid, client is granted access to the backend resources without having to go through authentication flow again.

F5 iRules:

```
when CLIENT_ACCEPTED {
        set authinsck 0
        set forceauth 1
        set ckname BIGXAUTH
        set ckpass 1xxx5678
        set ckvalue [IP::client_addr]
        set ckdomain .y.z
        set asid [AUTH::start pam default_radius]
}
when HTTP_REQUEST {
        if {[HTTP::cookie exists $ckname]} {
                HTTP::cookie decrypt $ckname $ckpass 128
                if {[HTTP::cookie value $ckname] eq $ckvalue} {
                        set forceauth 0
                }
                HTTP::cookie remove $ckname
        }
        if {$forceauth eq 1} {
                AUTH::username_credential $asid [HTTP::username]
                AUTH::password_credential $asid [HTTP::password]
                AUTH::authenticate $asid
                HTTP::collect
```

```
        }
}
when HTTP_RESPONSE {
        if {$authinsck eq 1} {
                HTTP::cookie insert name $ckname value $ckvalue path / doma
in $ckdomain
                HTTP::cookie secure $ckname enable
                HTTP::cookie encrypt $ckname $ckpass 128
        }
}
when AUTH_SUCCESS {
        if {$asid eq [AUTH::last_event_session_id]} {
                set authinsck 1
                HTTP::release
        }
}
when AUTH_FAILURE {
   if {$asid eq [AUTH::last_event_session_id]} {
           HTTP::respond 401 "WWW-Authenticate" "Basic realm=\"\""
   }
}
when AUTH_WANTCREDENTIAL {
   if {$asid eq [AUTH::last_event_session_id]} {
           HTTP::respond 401 "WWW-Authenticate" "Basic realm=\"\""
   }
}
when AUTH_ERROR {
   if {$asid eq [AUTH::last_event_session_id]} {
               HTTP::respond 401
   }
}
```

NetScaler Solution:

```
add lb vserver vip1 http 10.102.58.191 80

add authentication vserver vip2 ssl 10.102.58.192 443

add ssl certKey pi-srv -cert pi-srv.cert -key pi-srv.key

bind ssl vserver vip2 -certkeyName pi-srv

set lb vserver vip1 -authentication On -AuthenticationHost auth.mydomain.com

set authentication vserver vip2 -AuthenticationDomain mydomain.com -authentication yes

add authentication radiusaction radact -serverIP 10.102.61.134 -radKey freebsd

add authentication radiuspolicy radpol ns_true radact

bind authentication vserver vip2 -policy radpol
```

NetScaler allows you to configure and setup authentication parameters at vserver layer itself and it simplifies such common configuration tasks while performing application load balancing.