Selective Client Cert Authentication

Colin Walker, 2008-13-05

SSL encryption on the web is not a new concept to the general population of the internet. Those of us that frequent many websites per week (day, hour, minute, etc.) are quite used to making use of SSL encryption for security purposes. It's an accepted standard, and we're all fairly used to dealing with it in varied capacities. Whether it's that nifty yellow URL bar in Firefox, or the security warning saying that portions of the site you're going to are unencrypted, we've likely seen it before, and are comfortable with it in day to day operation.

What if, however, I wanted to get more out of my certificates? One of the more common, behind the scenes things that gets done with certificates is authentication. Via client-cert authentication users can have a "passwordless" user experience, automatic authentication into multiple apps with different access levels, and a smooth browsing experience with the applications in question. Combine these nice to have features with improved security, as it's much harder to spoof a client-cert than it is a password, and it's not surprising we're seeing a fair amount of companies putting this type of authentication into place.

That's all well and good, but what if you don't want your entire site to be authenticated this way? What if you only want users trying to access certain portions of the site to be required to present a valid client-cert? What's more, what if you need to pass along some of the information from the certificate to the back end application? Extracting things like the issuer, subject and version can be necessary in some of these situations. That's a fair amount of application layer overhead to put on your application servers - inspecting every client request, determining the intended location, negotiating client-cert authentication if necessary, passing that info on, etc. etc. Wouldn't it be nice if you could not only offload all of this overhead, but the management overhead of the setup as well? As is often the case, with iRules, you can.

With the below example iRule not only can you selectively require a certificate from the inbound users depending on, in this case the requested URI, but you can also extract valuable cert information from the client and insert it into HTTP headers to be passed back to the application servers for whatever processing needs they might have. This allows you to fine-tune the user experience of your application or site for those users who need access via client-cert authentication, but not affect those that don't. You can even custom define the actions for the iRule to take in the case that a user requests a URI that requires authentication, but doesn't have the appropriate cert.

There is a little configuration that needs to be done, like setting up a Client SSL profile to decrypt the SSL traffic coming in, but that should be simple enough. The iRule itself is pretty straight-forward. It uses the matchclass command to compare the URI to a list of known URIs that require authentication (class not shown in the example). If it finds a match, it uses the SSL commands to check for and require a certificate. Once this is found it uses the X509 commands to poll cert information and include it in some custom HTTP headers that the back end servers can look for.

```
when CLIENTSSL_CLIENTCERT {
  HTTP::release
  if { [SSL::cert count] < 1 } {
    reject
  }
}

when HTTP_REQUEST {
  if { [matchclass [HTTP::uri] starts_with $::requires_client_cert] } {
    if { [SSL::cert count] <= 0 } {
      HTTP::collect
      SSL::authenticate always
      SSL::authenticate depth 9
```

```
      SSL::cert mode require
      SSL::renegotiate
    }
  }
}

when HTTP_REQUEST_SEND {
  clientside {
    if { [SSL::cert count] > 0 } {
      HTTP::header insert "X-SSL-Session-ID"  [SSL::sessionid]
      HTTP::header insert "X-SSL-Client-Cert-Status" [X509::verify_cert_error_string [SSL::verify_res
      HTTP::header insert "X-SSL-Client-Cert-Subject" [X509::subject [SSL::cert 0]]
      HTTP::header insert "X-SSL-Client-Cert-Issuer" [X509::issuer [SSL::cert 0]]
    }
  }
}
```

As you can see there is a fair amount of room for further customization, as was partly mentioned above. Things like dealing with custom error pages or routing for requests that should require authentication but don't provide a cert, allowing different levels of access based on the cert information collected, etc. All in all this iRule represents a relatively simple solution to a more complex problem and does so in a manner that's easy to implement and maintain. That's the power of iRules, in a nutshell.

Get the Flash Player to see this player.

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc.
Corporate Headquarters
info@f5.com

F5 Networks
Asia-Pacific
apacinfo@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
emeainfo@f5.com

F5 Networks
Japan K.K.
f5j-info@f5.com