Unbind your LDAP servers with iRules

Colin Walker, 2008-09-10

LDAP is one of the most widely used authentication protocols around today. There are plenty of others, but LDAP is undeniably one of the big ones. It comes as no surprise then that we often hear different questions about using F5 technology with LDAP servers on the back-end. Whether people are looking for more performance, increased reliability and availability through load-balancing, or just more flexibility, there are many things that we can do to help.

A great example is improving performance. This example is driven from a client's requirements to reduce the overhead on their LDAP systems. They wanted to do so in a particular way, however. They were receiving a high-volume of very short-lived connections that all needed to query the back-end LDAP systems for information. Each one of these connections would open a new connection and as it turns out the overhead of setting up and tearing down the TCP connections was creating a fair amount of churn on their server, due to the high volume and short duration of the requests. Seeing this, they turned to their BIG-IP, looking for a solution.

As luck would have it, iRules was able to step in to help them accomplish just what they were looking for. Thanks to one of the many bright engineers here at F5, Nat Thirasuttakorn, they were able to leave server-side connections to the LDAP systems open for long periods of time and just manage the handshakes on the client side, thereby greatly reducing the overhead on the LDAP servers.

Below is the iRule that Nat was kind enough to share with me so I could pass it along to the DevCentral community. What it does is listen to the LDAP traffic, watching for an unbind to occur. Once the iRule sees an unbind from the client, which would normally be sent to the LDAP server terminating the connection, it simply uses the LB::detach command to detach the back-end connection at the BIG-IP, and tosses the unbind command itself so the LDAP server never sees it. This leaves the server's connection to the BIG-IP open and available for the next request that comes in.

It's important to understand that the LB::detach command isn't terminating any connection, even though the name might sound like it. All it's doing is detaching the current session from the connection it's established to the server in question, allowing future sessions to make use of the connection. This will, in essence, makes it look to the LDAP server that there's a single (or several), long-lived connection being held open with many requests flowing through. What's really happening is the BIG-IP brokering requests from the client and using already established back-end connections to keep overhead down to a minimum. This is the beauty of OneConnect (which is required for this solution to work) and iRules on the BIG-IP's TMOS architecture.

I don't have any performance numbers to share, but I'm willing to bet the BIG-IP is a fair amount more efficient at managing those large numbers of short-lived connections than an LDAP server is going to be. That means not only are you gaining the overhead back on your auth server, but you're not really losing much on the BIG-IP. That makes it a win-win. Again, many thanks to Nat for sharing the below code.

```
when CLIENT_ACCEPTED {
    TCP::collect
}
when CLIENT_DATA {
    binary scan [TCP::payload] xc ber_len
    if { $ber_len < 0 } {
        set ber_index [expr 2 + 128 + $ber_len]
    } else {
        set ber_index 2
    }
    # message id
    binary scan [TCP::payload] @${ber_index}xcI ber_len ber_len_ext
    if { $ber_len < 0 } {
        set ext_len [expr 128 + $ber_len]
        set ber_len [expr ($ber_len_ext & 0xffffffff) >>(4-$ext_len)*8)]
    } else {
        set ext_len 0
    }
    incr ber_index [expr 2 + $ext_len + $ber_len]
    # ldap message
    binary scan [TCP::payload] @${ber_index}c ber_type
    if { [expr $ber_type & 0x1f] == 2 } {
        log local0. "unbind => detach"
        TCP::payload replace 0 [TCP::payload length] ""
        LB::detach
    }
    TCP::release
    TCP::collect
}
```

Get the Flash Player to see this player.
20081009-ldap_no_unbind.mp3