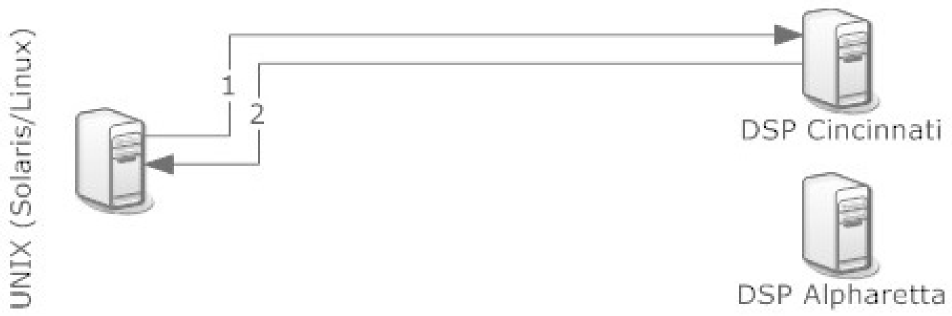
From: Spitzer, Andras ([redacted])
Sent: Monday, August 27, 2012 2:28 PM
To: [redacted] ([redacted])
Cc: [redacted] ([redacted]); [redacted] ([redacted]); [redacted] ([redacted]
[redacted]); [redacted] ([redacted]); [redacted] ([redacted]);
[redacted] ([redacted])
Subject: LDAP design proposal with F5

Paul,

Last Friday we discussed the possibility of using F5 in front of our LDAP Proxy servers. I decided to drew the concept so it's easier to understand and also makes it easier to involve others into the initiative. The pictures I drew are focusing only around the faster recovery concept using F5, hence it does not include all the details in our LDAP environment, it only covers the communication between the UNIX clients and the Directory Proxy Servers, it does not include nor recommends anything beyond the DSP.

1. **Today, normal operation**

Today, normal LDAP operation (<2sec)



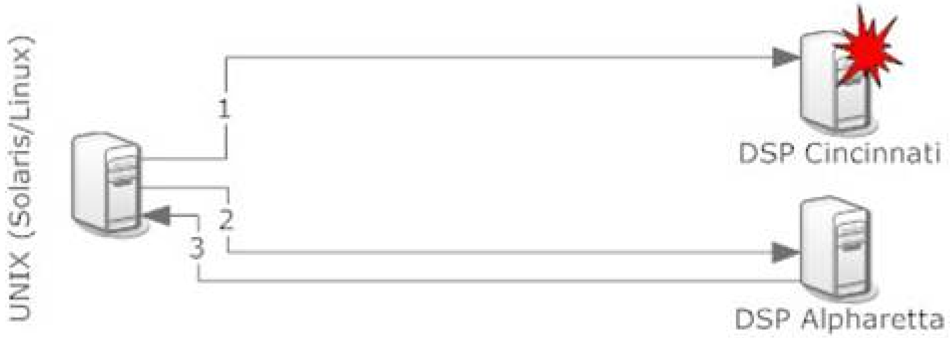
On the left side you see the LDAP client, which is our UNIX servers, can be any Solaris or Linux. On the right side, you see the two Directory Proxy servers, one in Cincinnati and one in Alpharetta. Today, our UNIX clients have two LDAP server IP address configured, these two IPs are the two DSPs. There is no load balancing on the UNIX side, the two IPs are used in a redundant fashion. If one of the IPs become unavailable, after a while the UNIX will start to use the other LDAP server IP address configured. This figure shows us how our UNIXes communicate with our LDAP servers under normal circumstances.

Step 1. UNIX host initiates an LDAP request
Step 2. Proxy server replies

That's the normal behavior, the LDAP communication usually **completes within 1-2 second**.

2. **Today, client based primary failover**

Today, client based failover (30sec > few minutes)



This section describes the currently in place failover mechanism, which is the client based failover. In this case the DSP (Directory Server Proxy) we communicate with stops responding. Now we have to differentiate two type of failures. The difference comes from the fact that LDAP is using TCP as the transport protocol, which is a reliable protocol by using segment sequencing and acknowledgements.

First type of a failure when the endpoint which experience issue on their side (which is the DSP Cincinnati in this case) can close the LDAP TCP connection by either initiating the "four way terminating" sequence with a FIN segment (clean disconnect) or by sending a RST segment (unclean disconnect). Both cases the LDAP client can immediately recognize that the LDAP connection is down, and will go to open a new one. This type of failure is the "luckier" one, as the LDAP client is notified about the disconnect, which can take action immediately, although this type of failure depends on that the OS on DSP Cincinnati is still healthy enough to send either a RST or FIN. One example of this type of failure when I shut down manually the DSP process on DSP Cincinnati, in which case the OS will cleanly initiate a disconnect toward all the LDAP clients connected to this OS.

Second type of a failure is worse, let's take the previous example, but for any reason the OS on DSP Cincinnati is not able to send any RST or FIN to the LDAP clients, or even worse the DS Proxy seems like it's working by accepting requests, still it does not answer them. This is the worse type of failure, as the UNIX clients are not receiving any notification about the problem, hence they are waiting, or in other words, hanging. The amount they'll hang depends on many things (TCP timers, LDAP timers, LDAP Library implementation, software bugs), but usually are between ten seconds up to few minutes. This type of failure is very unhealthy and can cause major issues across our environment, an example is the Oracle DB/VCS issue with SU we experienced lately.

It also worth mentioning that there are bugs too which can even further increase the delay such a timeout event, for example the JDK5/6 bug in the past when the java client was notified by a 15 seconds extra delay about the event of an unexpected LDAP disconnect.

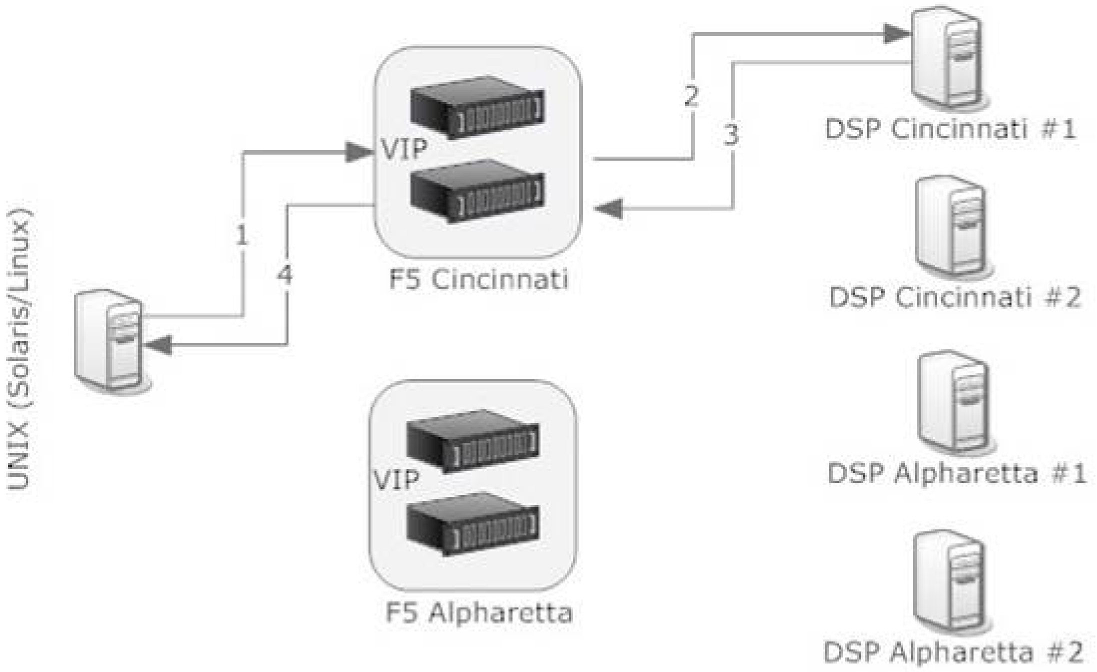
Now, as the figure shows :

Step 1. UNIX host initiates a request
Step 2. after various timeouts (TCP timeout, LDAP timeout, application logic timeout) as we didn't receive answer, we'll go the other LDAP server
Step 3. the other LDAP server replies

The **hang time** depends on whether we face type 1. or type 2. failure, but usually it's between a **few ten seconds up to a few minutes**.

3. Proposed, normal operation with F5

Proposed, normal operation with F5



This proposed change is to add a redundant F5 cluster between the LDAP clients and Directory Proxy servers. The F5s would have a Virtual IP (VIP) for the LDAP service, and the LDAP clients would have two IPs configured again, but this time the two VIPs instead of the two DSPs, providing a secondary failover option (slow) next to the primary failover option (fast) provided by the F5. In order to do that all the LDAP traffic have to go through the F5s, which can be done either with placing the LDAP servers "behind" the F5 so the LDAP servers would have the F5 as default gateway, or we can't/don't want to put the LDAP servers behind the F5 in which case the F5 will use SNAT to guarantee the LDAP servers will respond back via the F5. The advantage of the first option is that the LDAP servers logs will have valid client IP addresses, the disadvantage is additional configuration as we have to put the LDAP servers "behind" the F5. The second version is easier to configure as the LDAP servers can reside anywhere, but the LDAP logs will not contain the real client IPs, but the NAT'd IP from the F5. We should consider the pros/cons of each option, I personally prefer the first option.

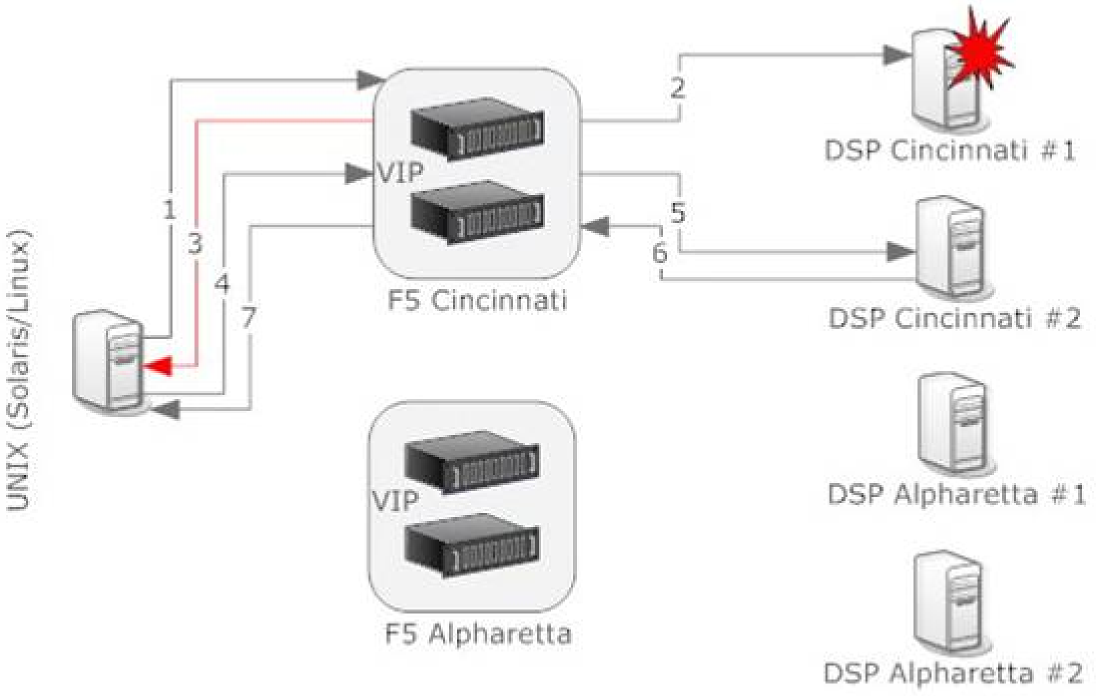
Anyway, the normal operation looks very similar to 1., although the F5 continuously monitoring the LDAP server running a test query, which if does not respond in a certain amount of time will declare the LDAP proxy as dead and removes immediately from the server pool. Also an addition that I would add a second DSP both in Alpharetta and Cincinnati for further redundancy, also the resource requirements for a new DSP is not that heavy (mainly network and CPU), as it does not hold any data it just directs traffic.

Step 1. UNIX host initiates a request
Step 2. F5 forwards the connection to the DSP
Step 3. DSP replies
Step 4. F5 forwards the response

F5 would not add any significant delay to the LDAP completion time, so normal operation would **complete somewhere between 1-2 seconds**, similar to our current case described at 1.

4. Proposed, F5 based primary failover

Proposed, F5 based primary failover (5-10secs)



Let's see how this proposed design works when we need to failover because the Directory Proxy server stops responding. We'll have a BIG-IP LDAP Monitor configured in the F5, we define the base DN, filter, and even we can define specific attributes we expect to see in the response. If the F5 won't receive the expected attributes from the DSP in a certain amount of time (5 seconds for example), it will mark the unresponsive LDAP Proxy as down, and the very smart move is that it will send a TCP RST flag back to all the UNIX hosts which had a persistent connection with this particular proxy, triggering the UNIX hosts to open a new connection with the F5, and the new LDAP connections will be forwarded to the working DSP only. This is a brilliant feature, as the UNIX hosts don't have to wait to figure out whether the LDAP server is down or not, the F5 will detect it, and more importantly will notify the UNIX hosts to open a new connection, which will go to the working Proxy as the Proxy which was marked down is automatically removed from the service pool.

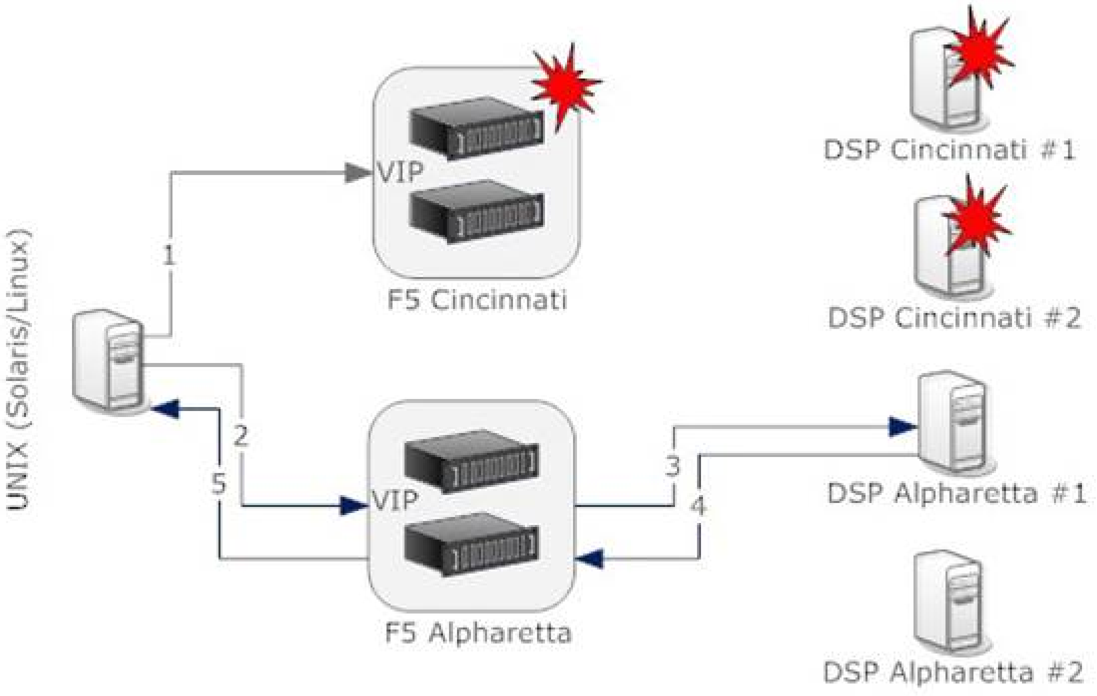
This is the reason why the F5 based failover can be way faster compared to the client based failover.

Step 1. UNIX host initiates a request
Step 2. BIG-IP LDAP Monitor realize that the DSP is not responding in 5 seconds (configurable), it marks the DSP down and automatically removes it from the service pool
Step 3. the F5 sends a TCP RST back to all the clients which had an active LDAP TCP connection with the DSP become unresponsive at Step 2., triggering all the UNIX hosts to open a new LDAP connection immediately
Step 4. UNIX host opens a new LDAP TCP connection
Step 5. F5 forwards the new connection request to the working DSP
Step 6. the working DSP replies
Step 7. F5 forwards the response

This failover mechanism depends on the BIG-IP LDAP Monitor timeout value we set, I recommend to run the monitor query every second, and the timeout should be around 5 sec. if we don't receive an answer for 5 sec for a simple query, the F5 should mark the DSP down and should notify the clients to open a new LDAP connection. The recovery time for this mechanism should be **somewhere between 5-10 seconds** (depends on the settings), and the best part is that the F5 notifies the UNIX hosts about the incident which can immediately open a new LDAP TCP connection to a working DSP.

5. Proposed, F5 based secondary failover

Proposed, F5 based secondary failover (30sec < few minutes)



In the proposed design we would still have two IP addresses configured in the UNIX host side for LDAP servers, although the two IPs would be the two F5 Virtual IPs. With this we could ensure that we have a second layer of redundancy which is even though recovers slower than the primary, but in case the F5 VIP becomes unreachable on one site, we can still switch to the F5 VIP on the other site. This scenario looks very similar to what we have today from the mechanism and recovery time point of view, although as this would kick off only when we'll face double failure within our environment, hopefully this would be very rare to experience.

Step 1. UNIX hosts initiates a request
Step 2. the request times out on the UNIX side, so it will connect to the other LDAP Server IP configured, which is the VIP on the site
Step 3. The F5 on the other site forwards the connection to the DSP
Step 4. DSP replies
Step 5. F5 forwards the response back to the UNIX host

The recovery time in this double failure scenario would be similar to what we have today, **between a few ten seconds and a few minutes**.

That's it, these scenarios show clearly the advantage we can earn using the F5 compared to what we have today. A few words still, even though using the F5 can help us to recover faster from an LDAP failure, we have to test and engineer thoroughly the LDAP service configuration within the F5 before we put it in production. For example the idle timeout can't be shorter on the F5 than it is in the LDAP server, we also have to make sure we set the idle timeout on both sides of the F5, the client and the server side as well, etc.

In case you are interested, the F5 LDAP configuration guide : http://www.f5.com/pdf/deployment-guides/ldap-iapp-dg.pdf

Regards,
sendai