Slingshot Server Multi-instancing

for Scalability and Reliability

Multiple instances of the Slingshot application server can run in parallel, being connected to the same database and to the same user file storage.

- ✓ *A Slingshot app server consists of a web server (IIS based) and a background task processor (a Windows service) installed on a same machine.*
- ✓ *By the "user files" meant are the files uploaded by app users to the server: e.g., document attachments, product images, export/import job data files.*

The web servers running on the instances (Slingshot server machines) can be exposed as a single web server with the help of a client request routing facility. It is recommended that the request routing is done at the HTTP level rather than at the TCP/IP one, specifically by the means of the Application Request Routing (ARR) extension of Microsoft Internet Information Server (IIS) installed on a separate server.

The sample environment described herein consists of a routing ("front-end") server, multiple Slingshot application servers (minimum two for the setup to make sense), a database server and a fileshare (exposed by an SMB file server under a name similar to \\corpnasan\sg2userfiles).

The routing server should host an IIS v. 7.0 or above with the Microsoft Application Request Routing (ARR) extension installed. ARR is a proxy-based routing module that forwards HTTP requests to application servers, based on HTTP headers, server variables, and load balance algorithms. This extension also provides a client affinity feature that maps a client to a particular server, controlled by the ARR, for the duration of the client session.

- ✓ *The client affinity is crucial since the Slingshot web application is "stateful", with the user session state kept in the server process memory. Consequently, the requests of a same client just have to be routed to a same Slingshot server for the client session to persist.*

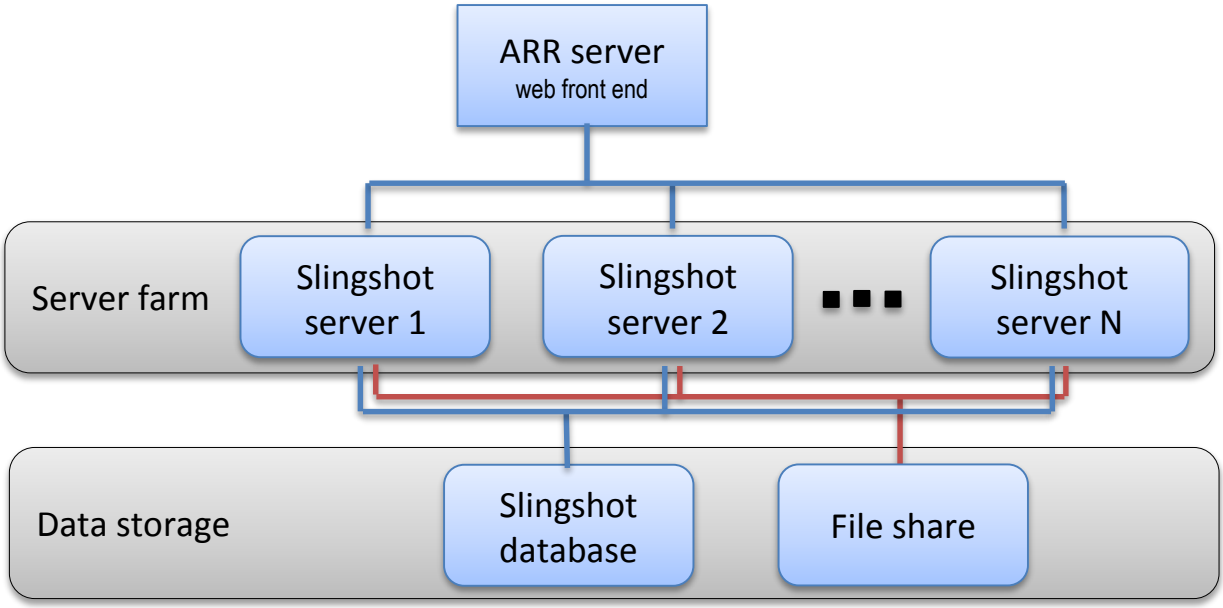
All instances of the Slingshot server should be configured similarly, with the critical similarities being all of the Slingshot apps connecting to a same Slingshot database and pointing to a same user files share, and the Slingshot web sites having been configured similarly (web app names, TCP port binding, authentication settings). An ARR configuration process involves the definition of a "server farm", the latter being essentially a named list of component web servers. To give an example, let our server farm be called SLINGSHOT, the Slingshot server machines constituting the farm be called SLINGSHOT1 and SLINGSHOT2, and the availability test URL for the Slingshot web app be http://everyserver/slingshot/test.aspx. With the help of the IIS Management console, the ARR makes it easy to add a new component server to the farm, or to remove an existing server from it (both operations take a few clicks). As a result, the Slingshot application server can be scaled up momentarily. To increase the application total capacity, a new Slingshot server machine should be set up, then added to the ARR server farm. That's it.

- ✓ *If you are using virtual machines (VMs), the simplest way to set up a new Slingshot server is to clone a Slingshot server existing VM.*
- ✓ *If you need to create a Slingshot server from scratch, it's no big deal too: install Windows, install Crystal Reports data engine (from an MSI), install Slingshot (with the help of the Slingshot Setup*

program). Please note that the Slingshot server instances don't have to be equally powerful. If they aren't, the servers capacity should simply be reflected in the ARR farm load balancing rule.

The data storage component of the Slingshot environment consists of a DBMS server and a fileshare, the latter exposed probably, but not necessarily by a NAS, or a SAN device. The DBMS server scalability and reliability is a responsibility of the organizational unit that hosts the Slingshot environment (including the DBMS). As to the scalability of the DBMS, it will likely be "vertical" – achieved by the adjustment of the DBMS host hardware. As to the reliability, it can be achieved by setting up a failover cluster (active/passive) of similar DBMS servers with data replication. The fileshare "scalability" and reliability can be achieved by properly configuring the storage device.

A Slingshot multi-instance environment possible configuration is illustrated with the diagram below:

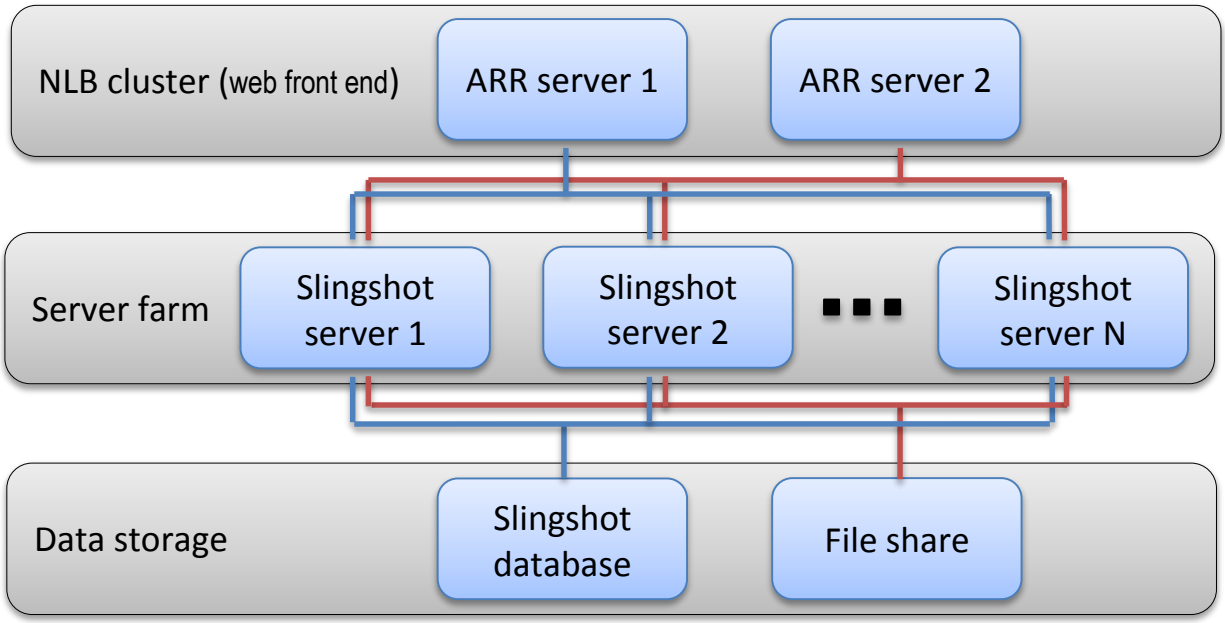


While the above architecture provides that the Slingshot multiple instances are exposed as a single web server, there are a few issues with it that may need to be addressed for the environment configuration to become perfect:

- The ARR server is a "single point of failure" in the above scheme.
- The scalability of the Slingshot application servers is limited by the capacity of the ARR server machine that acts as a web proxy.

To address these challenges, a "clusterization" of the very ARR server may be implemented. Multiple ARR physical servers (usually, two) may be joined into a cluster with the help of the Microsoft Network Load Balancing (NLB) feature. The cluster should be configured for the active/active operation to provide high availability and scalability. In this mode, both ARR servers will be forwarding the incoming requests, distributed between them by the NLB. The cluster setup process is simple. The NLB feature and management console should be installed on each of the ARR servers. A new cluster with a unique IP address (and network name) should be defined in the NLB console, then all of the ARR servers should be added to the cluster. The NLB cluster should be configured to accept traffic on all of the cluster nodes.

A multi-instance environment with a clusterized front end is illustrated with the diagram below:



Described above is a recommended configuration (with a singular or a clusterized ARR server).

- ✓ *It's been best in the application server failover tests.*

But there are other approaches for the app server "scaling", some of which might appear sufficient or preferable in your conditions:

In the cases, when the Slingshot application is accessed always from the IP address space, to which the Slingshot server belongs, advisable may be a simple solution with *no ARR layer*, where the request *routing is done by the NLB* installed on the very application server machines joined in an NLB cluster.

When the HTTP ARR is (still) preferable, but organizing it as a separate hardware level is not, available is the configuration where the Slingshot application and the ARR logical layers are implemented within the *same physical layer*. This approach is not recommended for an on premises implementation though, since it has limitations, while being more complex in setup. (Naturally, an NLB is also employed.)

- ✓ *An approach similar to the above supports the Slingshot Cloud multi-instancing in Azure (with no NLB since Azure has its own front end load balancer).*

The beauty of the above techniques is that is that the scalability is achieved via the "multiplication" of a Slingshot uniform entire server. This paper would not be complete though, if we didn't mention the approach based on the physical separation of the Slingshot server logical parts. To make it clear, the Slingshot web application and the Slingshot Background Processor service can be installed separately: each on its own machine. In fact, there may be any number of web server machines and any number of background processing ones. The former should be joined by the means of ARR or NLB so they act as a single web server, while the latter don't even have to be grouped in any way.

Slingshot specialists will be happy to answer your questions on this topic.