WhyNoWar

Why No War

Introduction

The documentation in 5.0 and later states that deployment in user-provided containers is no longer supported. This page is intended to explain this decision.

Solr is a server

Solr is intended to be a **server** not a **Java web application**, similar to mysql or the Apache web server. When Solr was first created, designing it as a web application was a convenient choice, to avoid writing a lot of tricky code to build a network layer. These days, this design decision has become a limiting factor.

When you download Solr and install it onto your machine, it should be **Solr** that gets started. It should not be necessary to start a completely different application (Jetty in this case) which then starts Solr.

At this time, Solr is still a webapp, but this is an internal implementation detail, not an immutable property. The intention is to make Solr into a completely standalone application. Startup scripts that start the included container are the first step towards that goal. Jetty might still be the technology used once Solr is a standalone application, but if that happens, it will be internally embedded.

Why remove a deployment option?

There is no single reason for the shift in design. The web application model has a number of limitations that the Solr developers fight with on a regular basis ... problems that wouldn't exist if Solr were a standalone application.

Here are some reasons for the decision:

- The binary server included with Solr, which is Jetty, along with the tuned server configuration that is also included, is used by a large portion of the test suite included in the Solr source code. If a user chooses to use a container from another source, especially if it's not Jetty, they are running in an untested environment, and it may not be possible at all to make it work.
- If everyone uses *only* the pieces that are shipped with the binary release, then they are running a fully tested system that will not vary from install to install.
  - Supporting a single set of binary bits is *FAR* easier than worrying about what kind of customized environment the user has chosen for their deployment.
  - If one person works out how to accomplish a task, they can publish that information in some way (blog, mailing list, etc.) ... and other users will be able to use it without any problems. When different users are using different servers, one user's step-by-step configuration or automation tools may not work for another user. One user may have features available in their container that are not available to other users using a different container.
- Right now all the network configuration is handled outside Solr in the container, but pieces necessary to interact with the network config are configured in Solr. All server configuration should be centralized in one place, not scattered between Solr and other software pieces.
- Solr does not know what port number the container is listening on until it receives a request from the outside. SolrCloud relies on knowing what address, port, and URL path each of its nodes is using. Currently that information must be fed to Solr via configuration that can be completely separate from the container config, which makes it possible to get the configuration **wrong** - configured with one setting in the container and configured with a different setting when SolrCloud registers with the cluster.
- If Solr owns the network layer, a large number of possibilities open up:
  - We can configure security in Solr *itself*. This is already becoming possible in current 5.x releases, but future changes will make this even easier.
  - The framework providing the network layer could be configurable. As an example, Zookeeper ships with one set of network bits active by default, but allows the user to choose to run with Netty providing network services.

The deployment options available in Solr 4.x encouraged people to deploy Solr in untested and unsupported ways, or to deploy Solr along with other web applications in the same container. Solr may not perform very well when the container is shared with other applications. Sometimes even sharing the *machi ne* with other applications can be problematic.

Other Related Changes

The webapp context path

The context path for Solr has been hardcoded to /solr since 5.0. Scripts that come with Solr assume this. The admin UI that became standard in 6.0 assumes this. Although you can change the configuration in Jetty to use a different context path, the admin UI and much of the scripting are going to stop working if you do. Fixing these problems will require manual editing of many different parts of the system. It is difficult to accommodate a user-configurable context path in all tools, so it is unlikely that this decision will be revisited. Additionally, when/if Solr becomes a standalone app, it is unlikely that the path will remain configurable.

Can I still deploy Solr 5.x or later in another container?

For the moment, probably, but we cannot be sure everything will work like it does in Jetty. You won't find documentation on how to do this here or in the reference guide. You'll need to consult the documentation for the container you plan to use.

Solr 5.0 through 5.2

There is still a solr.war file in the download, found in the server/webapps directory. If you take that solr.war file, the logging jars in server/lib/ext, the log4j.properties in server/resources, and the jetty context fragment in contexts, then you can deploy in a third-party servlet container just like you could with Solr 4.x.

Solr 5.3 through 7.3

Instead of a .war file, the Solr application is pre-extracted in the solr-webapp/webapp directory. Aside from changes in the location and the context configuration fragment, installation into another container would be similar to what you would do for previous 5.x versions that included a .war file.

Solr 7.4 and later

The big change in this version is a significant logging upgrade. Solr still uses the slf4j framework, but the final logging destination has been upgraded from log4j 1.2 to the most recent log4j2 version at the time that the work was done. See SOLR-7887 for detail. If you're going to deploy this version in another container, then the jars you need from server/lib/ext will be very different than in previous versions, and the logging config will be server/resources/log4j2.xml.

Information that's not version specific

Note that if you choose to do this kind of deployment, you are on your own. The project cannot support every container out there. Solr already has problems deploying in some containers, even with older releases that DID have the .war file in the dist directory. If bugs are filed on these problems, they will be closed without action.

At some point in the future, date unknown, Solr will become a completely standalone application. When that happens, there will be no guarantee that users can still compile Solr in a way that can be deployed in a third-party container. This remains an option at the moment, but that option is expected to disappear in a later release.

One of the ideas that has been considered is to make Solr into *two* applications – the first would be a controller application with minimal memory requirements. That application would be responsible for running and monitoring a second JVM running the main application. With proper communication between those two apps, we would have the ability to place more things under the control of the admin UI – things like security settings, listening port, memory allocations, GC tuning, Solr restarts, etc.

Running other applications in Solr's container

Possibly, but this is unsupported and not recommended. Solr ships with a completely standard Jetty that has been stripped down to only the components and configuration necessary to run Solr. It might be able to support other applications by adding additional configurations to the contexts directory. Some applications might require adding configuration and Jetty components that have been removed.

The reason that it's not recommended comes down to security. Giving end-users direct access to Solr is a major security issue. If users have direct access to other applications running in the same container, extra effort must be taken to ensure that Solr is secure. The general recommendation for Solr is to run it in a part of the network that unauthorized people cannot reach at all. If that recommendation is followed, there is no need to add any security.