AWS C libraries, the unknown heroes behind AWS tools and the adventure to package them

Date:	2024-07-10
modified:	2024-07-10
tags:	AWS, Fedora, EPEL, Packages, Packaging
description:	Packaging the AWS C libraries for Fedora and EPEL
category:	Linux
slug:	aws-c-libraries-the-unknown-heroes-behind-aws-tools-and-the-adventure-to-package-them
Author:	Dominik Wombacher
lang:	en
transid:	aws-c-libraries-the-unknown-heroes-behind-aws-tools-and-the-adventure-to-package-them
Status:	published

It all started with an orphaned Fedora package called **aws-php-sdk3**. I was interested to adopt and update it. I learned quickly that I have to take a look in the different dependencies too. I opened a can of worms, there were a few obvious dependencies and then some more hidden.

Like **aws-php-crt**, which isn't packaged for Fedora yet and upstream builds it in a way that it bundles a couple of AWS C libraries. As I learned in the meantime, bundling libraries or modules has to be avoided whenever possible based on the Fedora and EPEL packaging guidelines. You might get an exception but you need a good reason for it.

So I dig deeper and learned that there were some activities two years ago but except one of them, nothing went through and made it into Fedora. I spend a couple of days to understand what dependencies each library has. In which order do I need to build and package them to make them work. What was the state two years ago, what can be re-used.

After this exercise I had a full plan written down and started to create the packages in a copr project. That way I can satisfy the dependencies during development. Even if the packages are not in Fedora yet.

I learned that architecture *s390x* isn't supported by upstream (Archive: [1], [2]) and there is also no plan to change that. So this means every package needs an Arch Exception. In addition to that some packages had unit tests that require an internet connection. That's not given when Fedora packages are build, so I had to create patches so disable this kind of tests. Even if this means that the test coverage drops slightly. It's still better to run the majority of tests during package build to discover issues, instead of deactivating all of them.

With **s2n-tls** I hit an interesting problem that the tests were failing only on RHEL 9. Building on different Fedora versions and RHEL 8 was working without a problem. I reported the bug upstream (Archive: [1], [2]) and was at first unsure what makes RHEL 9 unique and causes the problems. After a bit of thinking and some feedback from the maintainers, I remembered that Red Hat announced a while back to deprecate SHA1 in their openSSL package (Archive: [1], [2]). So I did some research and came pretty close but couldn't track it down in the s2n-tls source. A maintainer provided a patch to verify my assumption. And indeed, the problem is that they use SHA1 certs in the unit tests. I have to admit it made me proud to help to identify this problem. I think they going to provide a fix in an upcoming release and then I can continue to package it for EPEL 9 :)

Right now I'm at roughtly 1/3, I packaged:

- aws-c-common
- aws-c-cal
- aws-c-sdkutils
- aws-c-compression
- aws-checksums

- s2n-tls

and they are at least available in Fedora *rawhide*. Most of them are already in all stable Fedora and EPEL branches.

Publishing happens with some delay because a new package needs ~7-8 days to make it into the stable repository. Which means, you start with one package were all other depend on. You go through the package review process, submit it, wait for a week. If you have other packages ready now. You do the same, you submit and wait. And so on and so on. Which means it takes overall around two to three months, depending on how fast the package reviews went through, till all AWS C libs are available.

Yes, there is the concept of Karma, but let's assume you not always find people that can invest the time. So the above assumption is based on the "worst case".

As you see, I was dragged away from my initial goal, packaging and keeping **aws-php-sdk3** alive, into a massive amount of groundwork. But as soon all AWS C libs are packaged and available, it opens the door for a lot of other AWS tools to be properly packaged without bundled libraries.

I'm really looking forward to that!