April 12, 2010

Subject: Comments regarding File Number S7-08-10

I am writing to submit some comments on a proposed SEC rule, with the following header information:

> SECURITIES AND EXCHANGE COMMISSION
> 17 CFR Parts 200, 229, 230, 232, 239, 240, 243 and 249
> Release Nos. 33-9117; 34-61858; File No. S7-08-10
> RIN 3235-AK37
> ASSET-BACKED SECURITIES

My name is J.D. Baldwin. I am a computer information security professional with over twenty years of experience in various areas of information technology. My academic qualifications include a master's degree in computer science from the University of Maryland, a CISSP certification, two certifications (intrusion analysis and InfoSec professional) from SANS GIAC and other, lesser certs. My work experience includes three years on the computer science faculty of the U.S. Naval Academy, research fellowships at Los Alamos National Laboratories and various system administration and network security positions at a large, multinational corporation.

I generally support the thrust of the proposal as I understand it. The intent of my comments is to address some technical details.

1. p. 18 of the proposal specifies that "The computer program would be tagged in XML and required to be filed with the Commission as an exhibit." Depending what you mean by "tagged in XML," this seems redundant and possibly even harmful. A computer program in any language already has a formal syntax. All XML tagging would do is make it more difficult to load the program into an interpreter, since the tags would have to be removed beforehand.

2. p. 216 asks the question explicitly: should Python be the standard language for the submission? I would suggest that Perl would make a better standard than Python. I have nothing whatever against Python, which I regard as a powerful and beautiful language. However, Perl is purely open source and vendor-neutral (as is Python). Perl has advantages over Python that are relevant to the SEC's purposes as I understand them:

- Perl has been around far longer than Python; the Perl community of developers is far larger, and their collective knowledge is far deeper, than is the case for Python
- There are far more "modules" (see below) available for Python, and the distribution infrastructure for Perl modules is far wider, more heavily reviewed, and more robust than for Python

On the subject of "modules" -- these are (relatively) small, limited-function packages of source code intended to provide special functionality in Perl. One installs a module on one's system, and then its functionality is available for use by all Perl programs on that system. Perl modules are (generally) publicly available and reviewed

by the entire user community, which tends to ensure their security and reliability. (All of this is true of Python as well, but again -- Python is not as widely used as Perl, and its advantages in this area are not as strong as Perl's.)

If this proposal goes forward, you should probably consider either providing your own modules, or specifying a set of modules that are acceptable for use. You will also have to specify a base minimum version of Perl or Python, as older versions have more limited module support, more limited "built-in" modules and more limited language features in general. This is true of both languages.

The above is the main point I wished to make to you. Here are some quick-impression answers to other points raised in the Request for Comments section of the proposal:

3. p. 217: "Should more than one programming language be allowed?" Probably not. Standardizing on a single, open-source language is your best bet for getting readable, usable code.

4. p. 217: "Should we restrict ourselves to only open source programming languages or allow fully commercial or partly-commercial languages (such as C-Sharp or Java) to be used? If so, what factors should be considered?" I feel strongly that the open-source requirement should be absolute. There is no advantage to be gained from permitting proprietary languages, and every disadvantage to doing so.

5. p. 217: "Under our proposal, issuers would be required to file the waterfall computer program in the form of downloadable source code on EDGAR. Prior to filing, the code would not be tested by the Commission. Would downloading the code onto a local computer give rise to any significant risks for investors? If so, please identify those risks and what steps or measures we should take to address the risks, if any." Any "foreign" program run on a system involves risk. You should warn users that such programs should never be run on production systems, or systems with "live" financial data. This is a common "best practice" and no competent system administrator would behave otherwise, but a warning to that effect would still be in order here.

I hope these comments are helpful to you. I am of course available to discuss these or any other aspect of the proposal you feel appropriate. The email address from which I am sending this is best and fastest for contacting me.

JD Baldwin