

Re: How regularly is the GnuPG source code examined?

Source: <http://www.derkeiler.com/Newsgroups/sci.crypt/2005-09/1241.html>

tomstdenis_at_gmail.com

Date: 09/27/05

Date: 27 Sep 2005 12:16:47 -0700

Alun Jones wrote:

- > *Good tools help that, though. Marking return values as "must be checked",*
- > *and then checking that all "must be checked" return values _are_ checked is*
- > *a must-have on a source analysis tool.*

There are many things to do. Usual good first steps are to check C compiler warnings. Then static tool warnings [e.g. splint]. After that you have to actually manually verify each function.

That is for any function, prove that it produces the results you want and ONLY the results you want [or define]. This is called verification and is completely different from testing.

Think of testing as more of a black box. I put in good_sample_x and get good_response_y, I put in bad_sample_x and get bad_response_y.

That tells us if the code is **working** but not if it's correct. Bugs occur when code is not correct and often in code that is working.

- > *Clear commenting and structure, while they are a hindrance to sheer speed of*
- > *development, greatly aid in reading the code and deciphering its function.*
- > *Obviously, the code is the code, and the comments aren't, so you'll*
- > *occasionally get something like:*
- >
- > */* Add 5 to x */*
- > *x*=5;*
- >
- > *OR*

<snip>

Agreed. Definitely good coding practices is a required solid first step. It's not sufficient nor is testing.

As far as I know no open source project has EVER gone through a proper verification cycle. In many cases this is ok because a bug or failure

sci.crypt: Re: How regularly is the GnuPG source code examined?

is not harmful, just annoying. But in the case of cryptography it can be a nightmare.

Tom