

Re: Hardening FreeBSD, does anyone have any documentation that may help?

Re: Hardening FreeBSD, does anyone have any documentation that may help?

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/hackers/2006-11/msg00215.html>

- *From:* Jeremie Le Hen <jeremie@xxxxxxxxxx>
 - *Date:* Tue, 21 Nov 2006 16:09:28 +0100
-

Robert, Joerg,

thank you for replying.

On Tue, Nov 21, 2006 at 03:19:58PM +0100, Joerg Sonnenberger wrote:

On Tue, Nov 21, 2006 at 11:59:27AM +0000, Robert Watson wrote:

FYI, Silby gave a nice mini-talk/discussion at EuroBSDCon on the topic of gcc4 security features. It seems like there's a lot of support for having these things in FreeBSD, but a strong reluctance to have large outstanding patchsets against the compiler and build chain, hence the continued "strategy" of waiting for them to arrive in gcc4. Most questions boiled down to:

I fully understand this reluctance and that's why I am struggling to maintain a patch applicable to the latest RELENG_6 and CURRENT.

– What are the ABI impacts? Assuming that protection features arrive and depart, and that reasonable application backward compatibility is required for programs and libraries. Of particular interest was the case where we turn on a protection feature in X.Y and discover that this was a bad idea, so turn it off in X.Y+1.

The ABI impact is limited to the stack guard cookie, the initialisation function and the failure handler. Three different solutions can be used:

- (1) The code can be part of a separate library (libssp).
- (2) The code can be part of libc (DragonFly, OpenBSD and glibc do this).
- (3) Like (2), but the cookie is part of the Thread Control Block, e.g. accessible via %gs. This is done on newer glibc systems and has the advantage of avoiding PIC references.

Concerning backward compatibility, there are two cases.

Re: Hardening FreeBSD, does anyone have any documentation that may help?

Re: Hardening FreeBSD, does anyone have any documentation that may help?

The first one is ProPolice being integrated to the source tree (I mean the GCC stuff as well as the SSP symbols in libc). Turning the feature on or off is just the matter of using a compilation flag (toggled by some build knob). Since the SSP symbols are in libc at any rate, you won't have any problem to go back to a non-SSP world. Even SSP-protected third-party applications will simply depend on SSP symbols in libc until they are recompiled.

The second one is ProPolice being maintained as an external patchset as it is currently. This is trickier because the user may want to remove the patch and will thereafter have a libc without SSP-symbols, breaking **all** applications requiring them. Even installworld will break in this case, since libc.so is installed quite early. I've created libssp.so that contains SSP symbols, so users can temporarily circumvent this problem with LD_PRELOAD. This is described in the FAQ available on my website.

– What are the performance characteristics in a variety of real-world workloads?

The original benchmarks done with Propolice by IBM suggest typical degradations in the area of 2%–5%, depending on how many functions are called and not inlined and how many of them need to get the protection. The site of Etoh has more details.

When I release this ProPolice patch, I made a classical benchmark, building world thrice (available on my website as well):

```
WITHOUT_SSP WITH_SSP Overhead (%)
real 319m45.465s 323m31.551s 1.1%
user 180m5.823s 182m48.844s 1.5%
sys 23m18.855s 23m50.322s 2.2%
```

--

Jeremie Le Hen

<jeremie at le-hen dot org ><ttz at chchile dot org >

freebsd-hackers@xxxxxxxxxxxxx mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-hackers>

To unsubscribe, send any mail to "freebsd-hackers-unsubscribe@xxxxxxxxxxxxx"

Re: Hardening FreeBSD, does anyone have any documentation that may help?