

## Re: FreeBSD Patch question

**Source:** <http://www.derkeiler.com/Mailing-Lists/FreeBSD-Security/2003-09/0244.html>

---

**From:** Robert Watson ([rwatson\\_at\\_freebsd.org](mailto:rwatson_at_freebsd.org))

**Date:** 09/25/03

Date: Thu, 25 Sep 2003 16:13:30 -0400 (EDT)

To: "V. Jones" <[vjones62@earthlink.net](mailto:vjones62@earthlink.net)>

On Thu, 25 Sep 2003, V. Jones wrote:

> *I administer a remote server and want to apply some of the security  
> patches. (I assume this is the best way to go since I can't go into  
> single-user mode to use CVsup).*

I generally follow the following practice:

```
cvsup in multiuser
buildworld in multiuser
buildkernel in multiuser
```

These stages, other than impact on cpu, memory, disk i/o speed, and storage space, shouldn't interact with the running environment and so shouldn't be a problem. Then comes the slightly more tricky bit: I decide whether I'm willing to update while running multiuser. If I am:

```
installkernel
reboot
installworld
mergemaster
reboot
```

If I'm not, the procedure is much the same except that I boot only to single-user after the first reboot, mount -a, swapon, and proceed.

Note that there are a number of risks and complications associated with the installworld and mergemaster steps, both in multiuser and singleuser mode. multiuser is typically more risky: for example, if mergemaster notices a change to MAKEDEV, it will prompt to recreate devices. **DO NOT DO THIS ON A LIVE MULTIUSER SYSTEM.** :-) If you do run it, it will reset all the permissions in /dev, leaving in-use ttys world readable and writable. This will permit unprivileged users to potentially sniff the I/O for more privileged users, send output to their display, etc. So it's fine in single-user, but not multi-user. Typically, that sort of change doesn't occur on the security/release branches, but will happen with

## FreeBSD–Security: Re: FreeBSD Patch question

moderate frequency as you track –STABLE.

> *I have a couple of questions. First, I have installed one of the pgp  
> ports to verify the patches. When I run it, I get this message:*  
>  
>> *File 'buffer46.patch.asc' has signature, but with no text.*  
>> *Text is assumed to be in file 'buffer46.patch'.*  
>> *signature not checked.*  
>> *Signature made 2003/09/17 18:02 GMT*  
>> *key does not meet validity threshold.*  
>  
>> *WARNING: Because this public key is not certified with a trusted  
>> signature, it is not known with high confidence that this public key  
>> actually belongs to: "(KeyID: 0xCA6CDFB2)".*  
>  
> *I guess that I need to do some additional set up to get pgp to validate  
> this file. Can anyone tell me where to find a howto on this subject or  
> tell me what to do?*

PGP relies on a "web of trust". Users sign each others identities to bind them to keys. Your local PGP keyring will hold any keys and signatures you've stuffed in there. PGP determines "trust" by building a path of signatures and validations between you and the target key. There are various parameters to determine the degree of transitivity to trust, etc. There's fairly extensive documentation of the PGP trust model on various web pages, but you can read the above warning as simply "There is no path of signatures between your trusted keys and the key used to sign this message/file". For the highest level of confidence, attend a USENIX or BSDCon key signing, and sign the security–officer key yourself once you've seen the fingerprint, etc. For lower levels of confidence, go to a key–signing event with someone who has signed the security–officer key, etc, etc.

> *Second, Do I have apply each patch, then run make after each patch, or  
> can I apply all the patches and just run make once?*

It depends a bit on the patches and the branch. If you're tracking a release/patch branch, you can cvsup forward to the head of the branch, then rebuild the identified components. Sometimes, patches and update activities coalesce well (unrelated change to unrelated binaries). Sometimes, less well — you might have to make sure to build libraries before binaries, for example, or apply a series of sendmail or ssh patches in order. Cvsuping forward and rebuilding world and kernel is a reasonable answer for most people, and means you don't have to worry about the ordering.

FYI, regarding your general interest in advice: the single best piece of advice for remotely administered systems is to get a serial console. That way if something gets messed up, you have a decent chance of being able to fix it. It means you have full access to single–user mode, you can select which kernel to run at boot, even have multiple root file systems

## FreeBSD-Security: Re: FreeBSD Patch question

(production, backup) and swap between them. It takes a lot of the risk out of upgrades by providing a good escape route if networking fails to come up properly, for example. With the recent ARP fix, there was a functional regression in the first version of the patch, which caused routing to fail under some circumstances. If you had access to a serial console for a remote box, you were fine because you could revert to the previous kernel once you noticed the problem. Otherwise, you might be out of luck...

Robert N M Watson FreeBSD Core Team, TrustedBSD Projects  
robert@fledge.watson.org Network Associates Laboratories

---

freebsd-security@freebsd.org mailing list

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

To unsubscribe, send any mail to "freebsd-security-unsubscribe@freebsd.org"