Subject: Re: TinyVZ 0.7 released
Posted by samiam on Mon, 22 Aug 2011 18:35:35 GMT
View Forum Message <> Reply to Message

> This is very nice. My concern, though, is that things such as uClibc
> were not built with security in mind. I am pretty sure that uClibc is
> problematic when used in conjunction with SUID/SGID programs.

You know, this is going to make me start yet another almost completely off-topic rant.

Bruce Schneier summed it up best when he said that security is a process, not a product. In other words, security is a process of discovering how to make programs more secure. In the 1980s, programs were so insecure, it was not unheard of for programs to have intentional backdoors in them (the Sendmail "Wiz" command, for example). Soon break-ins were not a simple matter of finding backdoors (or default accounts, such as what the 1980s antagonist in Cliff Stoll's "The Cuckoo's Egg" used); people had to use buffer overflows, which a lot of 1990s code had.

C programmers have cleaned up their act; not only do skilled programmers have coding styles which avoid these kinds of mistakes, but there are a lot of other tricks to minimize the impact of this kind of error (the NX bit, etc.)

Not even the most skilled programmer in the world is guaranteed to make a perfectly secure program. For a while, advocates of Dr. Bernstein's software believed his software was perfectly secure and never needed to be updated, but while he is a brilliant programmer who makes very few security mistakes in his code, there is still the occasional security bug: djbdns, for example, has three known security bugs. [1]

Bernstein's software has a powerful lesson: For a program to remain secure, it has to be maintained. There has to be someone who is accountable for the security holes in the software and is willing to fix it. This is why I use a derivative of RedHat enterprise Linux: RedHat stands behind their software for seven years and I know I can keep a system reasonably secure for that time duration without having to be on the constant update-by-reinstall treadmill most other Linux distributions keep me on.

One of the reasons I have kept the number of packages in my TinyVZ distribution down to an absolute minimum is because this minimizes the number of potential security holes I will have to babysit in this distribution.

> Does uClibc ensure that fd 0-2 are open on program startup (opening them to
> /dev/null / /dev/full if not)? I doubt it. I admit I haven't checked,
> though.

It might. It might now. I haven't checked either. What I can tell
you is that there does not appear to be any vulnerability reports for
uClibc:

 http://security-tracker.debian.org/tracker/source-package/uc libc

(note to self: There is a vulnerability report for the gzip stuff in
the Busybox included with TinyVZ)

> I think a better libc to use for this purpose would be musl:
>
> http://www.etalabs.net/musl/
>
> It also lacks some of those highly desirable security measures, but I
> think it will gain those soon.

If I were to do this again in a few months, I may use musl; however
musl is an "alpha" product while uClibc is a mature product that is
still being updated. I use the code which works today; that's why I'm
using OpenVZ and not, say, LXC [2].

TinyVZ is, in truth, me putting closure on a project I had back in
2007 (around the time I started rewriting MaraDNS's recusrive
resolver) making a tiny uClibc + Busybox live CD distribution for
having on a business card CD so I could use cyber cafes and friends'
infected computers in a reasonably secure manner.

I never distributed that code, mainly because I never fully made it
independent from the DeLi distribution it was built on until, by a
weekend of hard-core hacking, I made the system self-hosting about a
week ago. From there, it was relatively simple to add Bash and write
the scripts used by OpenVZ to configure the system with vzctl.

> Meanwhile, the sad truth may be that under Linux we need to use (e)glibc
> (or other clones of it) for SUIDs/SGIDs.

This can very well be true. One thing I have done is minimize the
attack surface with SUIDs by having only two SUID programs in the
system: "passwd" and "su". While Busybox is supposed to have a way of
having it be SUID and drop privileges as needed, I don't fully trust
that mechanism; better to compile Busybox twice.

><plug>BTW, the full Owl
> userland, with development/build tools, is just 112 MB under .tar.gz:

> http://mirrors.kernel.org/openwall/Owl/current/vztemplate/
> It is also able to rebuild itself, although the source code is not part
> of the .tar.gz (just the development/build tools/libs are). With the
> source tarballs added, the size increases a lot indeed... to 280 MB if
> we exclude just the Linux kernel, which is not installed in an OpenVZ
> container anyway.
> </plug>

With the full self-hosting development tree being just over 40 megs xz compressed, and a usable "DNS toaster" system (which is resolving all of my DNS queries as I type this) being about 2 megs in size, I think TinyVZ targets those who want to have a really small OpenVZ system. OpenVZ's strength is that it allows a single computer to safely run a dozen or more separate services with full compartmentalization. By making the containers as small as possible, I can visualize a single server rack with a hub inside of it, as well as a dozen or so credit-card sized computers with Atom SOC cpus, 4 gigs of memory, and 64GB SSDs. Each one of those computers can run dozens of really tiny OpenVZ single-task containers.

Owl looks like a really good distribution, and I think it is very wise to stay in step with RedHat, since then RedHat's security updates can be applied. Does OWL have its own mechanism for applying security patches, or does it just use the patches from CentOS or Scientific Linux [3]?

> Alexander

- Sam

[1] I have blogged about this, see http://aac2.vk.tj

[2] Idiots who say OpenVZ is "deprecated" and insist LXC is the future of Linux containers really annoy me. LXC has not had the extensive security audit OpenVZ has (yes, a lot of LXC's code was contributed by the OpenVZ developers). Changing a solution that works today and that the development team stands behind with constant security updates with one that is new and unproven based upon baseless accusations of it being "deprecated" is not, IMHO, a good idea. Examples: http://ahva.vk.tj http://ahvb.vk.tj

[3] I have blogged about how it is annoying how security updates sometimes aren't made available to CentOS systems: http://acs2.vk.tj. I have also blogged about how to apply Scientific Linux security updates to running CentOS 5 systems, since there isn't a "Scientific Linux 5" OpenVZ template: http://ahi2.vk.tj.