Put a process in a sandbox where it can do least harm

Asked 16 years, 2 months ago Modified 13 years, 1 month ago

Viewed 6k times

25

I'm looking for the concept to **spawn a process** such that:

- it has only access to certain libraries/APIs
- it cannot acess the file system or only specific parts
- it can **do least harm should malicious code run in it**

This concept is known as sandbox or jail.

It is required to do this **for each major Operating system (Windows, MacOSX and Linux)** and the question is conceptual (as in what to do, **which APIs to use and and what to observe**) rather then language specific.

answer requirements

I **really** want to accept an answer and give you 20 points for that. I cannot accept my own answer, and I don't have it yet anyway. So if you **really** want your answer to be accepted, please observe:

- The answer has to be specific and complete
- With specific I mean that it is more then a pointer to some resource on the internet. It has to summarize what the resource says about the topic at least.
- It may or may not contain example code, but if it does please write it in C
- I cannot accept an answer that is 2/3 complete even if the 2/3 that are there are perfect.

this question FAQ

- Is this homework? No.
- Why do you ask this like a homework question? If you ask a specific question and you want to get a specific answer, and you know how that answer should look like, even though you don't know *the* answer, that's the style of question you get.
- If you know how it should look like, why do you ask? 1) because I don't know all the answer 2) because on the internet there's no single place that contains all the details to this question in one place. Please also read the stackoverflow FAQ
- Why is the main part of your question how to answer this question? Because nobody reads the FAQ.

windows linux security language-agnostic macos

Share

Improve this question

Follow

edited Sep 26, 2008 at 11:00

community wiki
8 revs
Florian Bösch

9 Answers

Sorted by: Highest score (default)

15

Mac OS X has a sandbox facility code-named Seatbelt. The public API for it is documented in the sandbox(7), sandbox_init(3), and related manual pages. The public API is somewhat limited, but the facility itself is very powerful. While the public API only lets you choose from some pre-defined sandboxes (e.g. “All sockets-based networking is prohibited”), you can also use the more powerful underlying implementation which allows you to specify exactly what operating system resources are available via a Scheme-like language. For example, here is an excerpt of the sandbox used for portmap:

```
(allow process-exec (regex
#"^/usr/sbin/portmap$"))
(allow file-read-data file-read-metadata (regex
    #"^/etc"
    #"^/usr/lib/.*\.dylib$"
    #"^/var"
    #"^/private/var/db/dyld/"
    #"^/dev/urandom$"))
(allow file-write-data (regex
    #"^/dev/dtracehelper$"))
```

You can see many sandboxes used by the system in /usr/share/sandbox. It is easy to experiment with sandboxes by using the sandbox-exec(1) command.

For Windows, you may want to have a look at David LeBlanc's "Practical Sandboxing" talk given at Black Hat USA 2007. Windows has no built-in sandboxing technology per se, so the techniques described leverage an incomplete mechanism introduced with Windows 2000 called SAFER. By using restricted tokens, one can create a process that has limited access to operating system resources.

For Linux, you might investigate the complicated SELinux mechanism: SELinux home, a HOWTO. It is used by Red Hat, for example, to harden some system services in some of their products.

Share Improve this answer Follow

answered Sep 30, 2008 at 13:52
Trance Diviner
319 ● 1 ● 3

Enlightening. I did not know of 'sandbox' on OS X. – ayaz Dec 27, 2008 at 8:52

8

For Windows there is a sandbox in Google Chrome. You may want to investigate it. It uses liberal BSD-like license.

For Linux there would be good old chroot or more sophisticated http://plash.beasts.org/wiki/.

OS X since Leopard has some SELinux-like protection available.

Share Improve this answer Follow

answered Sep 26, 2008 at 8:43

Paweł Hajdan
18.5k ● 9 ● 52 ● 65

There are several ways to escape from badly built chroots, once a process has gained root, e.g. by mknod+mount of the root file system. To really lock down a process under Linux use SE-Linux. – David Schmitt Dec 27, 2008 at 9:40

5

The site codepad.prg has a good "About" page on how they safely allow the execution of any code snippets..

> Code execution is handled by a supervisor based on geordi. The strategy is to run everything under ptrace, with many system calls disallowed or ignored. Compilers and final executables are both executed in a chroot jail, with strict resource limits. The supervisor is written in Haskell.
>
> When your app is remote code execution, you have to expect security problems. Rather than rely on just the chroot and ptrace supervisor, I've taken some additional precautions:
>
> - The supervisor processes run on virtual machines, which are firewalled such that

> they are incapable of making outgoing connections.
>
> - The machines that run the virtual machines are also heavily firewalled, and restored from their source images periodically.

Share Improve this answer Follow

answered Dec 27, 2008 at 3:18

community wiki
dbr

3

FreeBSD has specific concepts of jails, and Solaris has containers. Depending on what you're looking for, these may help.

chroot jails can help to limit what an application can do (though any app with root privileges can escape a jail), and they're available on most UNIXen, including OS X.

As for Windows, I'm not sure. If there was an easy way to sandbox a Windows app, most of them would be a lot more secure by now, I'm sure.

Share Improve this answer Follow

answered Sep 25, 2008 at 20:29

Dan Udey
2,977 ● 2 ● 21 ● 18

2

On windows (2000 and later) you can use Job objects to restrict processes.

Share Improve this answer Follow

answered Sep 27, 2008 at 2:07

Anders
101k ● 15 ● 120 ● 172

2

If you really want a technique that will work with all these platforms, as opposed to a separate solution for each platform, then I think your only answer is to set up a virtual machine for each testing environment. You can restore back to a snapshot at any time.

Another big advantage of using virtualization is that you can have all of the testing environments with their guest operating systems all on the same box.

Share Improve this answer Follow

answered Dec 27, 2008 at 7:22

community wiki
dj_segfault

1

For Linux, there is AppArmor. Unfortunately, the project is somewhat on hiatus.

Another sandboxing-alternative is VServer, which uses virtualization.

Share Improve this answer Follow

answered Sep 26, 2008 at 8:50

Torsten Marek
86.3k ● 21 ● 93 ● 98

0

Generally any virtual private server will do:

Linux VServer http://linux-vserver.org/Welcome_to_Linux-VServer.org

Parallels Virtuozzo Containers http://www.parallels.com/products/pvc/

and as was mentioned FreeBSD and Solaris has own implementations.

Oh. actually I've noticed you're asking it to work on ANY OS. Well, that might be complicated a bit as the I think less effort is just to reuse some VM that can support some level of sandboxing like:

- Java
- .NET

Share Improve this answer Follow

answered Nov 11, 2011 at 23:55

community wiki
NothingLikeGuru

-1

I'm not an expert on the topic, but i think the standard answer for linux is to define a SeLinux policy with the right capabilities for the process.

Share Improve this answer Follow

answered Dec 27, 2008 at 3:27

community wiki
TokenMacGuy

1 That's not what SELinux is about. SELinux is an execution auditing framework, i.e. it keeps a watch on processes, if they do only, what they're permitted to do, on a global level (including root), and if a process misbehaves it's being terminated and reported. Also it disallows execution of binaries that are not signed and registered in the list of what's allowed to execute. – datenwolf Dec 23, 2020 at 12:44