Hentzenwerke Whitepaper Series

When X Dies: How to Kill and Restart X Windows

By Whil Hentzen

A GUI is a complex piece of software; combined with the event driven capabilities, you may run into the situation where your GUI locks up on you. Unlike Windows, the Linux GUI isn't tightly bound into the underlying OS; instead, it's a set of separate programs running on top of the OS. As a result, however, you're not forced to restart your machine due to problems with the GUI itself. This whitepaper discusses the options you've got available to you to recover a failed X Windows session.

Hentzenwerke Publishing, Inc. • articles@hentzenwerke.com • www.hentzenwerke.com

1. Preface

1.1 Copyright

1.2 Revisions

1.2.1 History

Version	Date	Synopsis	Author
1.0.0	2005/1/10	Original	WH
1.1.0	2006/10/22	Added unresponsive keyboard tips	WH

1.2.2 New version

The newest version of this document will be found at www.hentzenwerke.com.

1.2.3 Feedback and corrections

If you have questions, comments, or corrections about this document, please feel free to email me at 'articles@hentzenwerke.com'. I also welcome suggestions for passages you find unclear.

1.3 References and acknowledgments

Thanks to MLUG members Daniel J. Cody, Jerry Davis, Jonathan C. Detert, Tom Francis, Darrick Hartman, Mark Pinkerton, and Michael Stravs for various tips and tricks, and for not laughing when I asked dumb questions.

1.4 Disclaimer

No warranty! This material is provided as is, with no warranty of fitness for any particular purpose. Use the concepts, examples and other content at your own risk. There may be errors and inaccuracies that in some configurations may be damaging to your system. The author(s) disavows all liability for the contents of this document.

Before making any changes to your system, ensure that you have backups and other resources to restore the system to its state before making those changes.

All copyrights are held by their respective owners, unless specifically noted otherwise. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark. Naming of particular products or brands should not be seen as endorsements.

1.5 Prerequisites

This document was written using Red Hat 8 and Fedora Core 3, and assumes a beginner's familiarity with use of Linux via the GUI and the Command Window.

2. Genesis: What happened to lock up the GUI

“I didn't touch a thing! It just stopped working by itself!” How many times have you heard that? Well, this time it's the truth, really, it was.

I had a GUI on a Red Hat 8 box, and was poking around on it, trying to add some tools that I've used for a while on Fedora Core releases. After I added the Show Desktop and X Screensaver buttons on the panel, the desktop and panel locked up. I could still move the mouse, but clicking on the desktop icons, right-clicking on the desktop to attempt to bring up a context menu, or clicking on the panel were all unresponsive. I did have a Konqueror window open that was still working, but that was it.

Having had 15 years of Windows experience, I knew I could simply restart the box and be all set again, and with Linux's file system robustness, I wasn't worried about losing data. But the box had been up for the better part of a year and I saw no reason to bring it down just to restart the GUI. It's just a program, after all.

I searched the Internet for options, and while there was a lot of advice out there, some of it conflicted. In my experience, I've also seen the situation where, once in a while, someone offers a suggestion that's well-intentioned but not completely accurate. Furthermore, nearly all of it was written badly. So I turned to my local Linux User Group, MLUG, and described the solutions I'd found, and asked how to sort the advice out.

Here's what I've found.

3. Dealing with an unresponsive keyboard

The rest of this article assumes that you can use the keyboard. But what if the keyboard is unresponsive in X – you can't even toggle CapsLock or NumLock, much less do something useful like switch to a console?

Michael Stravs suggests a system request to give keyboard control back to the console: Alt-SysRq-r. (SysRq is the same as PrntScrn.) Then you can switch to the console with Ctrl-Alt-F1.

Additionally, if you **really** want to reboot, before hitting the reset button, he suggests trying Alt-SysRq-s in order to ask Linux to flush all buffers to disk. Then reboot with Alt-SysRq-b (reboot).

4. Ctrl-Alt-Backspace

The overwhelming choice was to try Ctrl-Alt-Backspace first. By definition, this keystroke combination will explicitly restart the X server.

Tom Francis said, "If you're able to type into one of your windows, the basic X server functionality is still working, and it's your window manager (or possibly session manager) that's messed up. You can try to kill and restart the window (or session) manager on your own, but that's usually not easy.

When the X server dies, it will send TERM and then KILL to all processes using the X server, meaning all of your windows, window manager, and session manager.

Also, if you were using a login manager, that manager would notice that the X server isn't running and restart it, giving you a new login screen.

5. Working with a console if X is dead

If Ctrl-Alt-Backspace didn't kill the X Server, then you'll probably need to try other tasks, but if you can't use the panel to open a terminal window, you've entered the land of 'chicken and egg' - you need to open a console to kill X, but since X is dead, you can't use the GUI to open a console.

Fortunately, there are other options available to open a console to work with a console window.

In normal configurations (unless you've messed with the configuration of your system), Ctrl-Alt-F1 will open a full screen console, as shown in Figure 1.

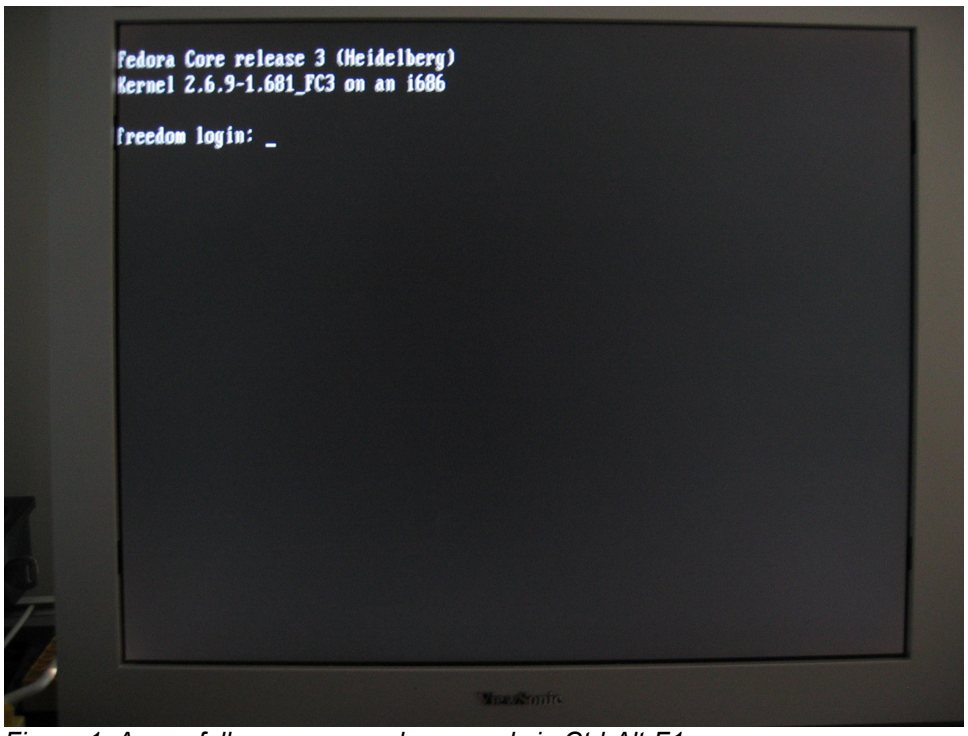

Figure 1. A new full-screen console opened via Ctrl-Alt-F1.

The console display is the same one as appears if you boot up in run level 3. You'll be prompted to log in as a user on the system, Once you've logged in, you can run commands and programs as needed, just as if you were in a console window in the GUI, as shown in Figure 2.

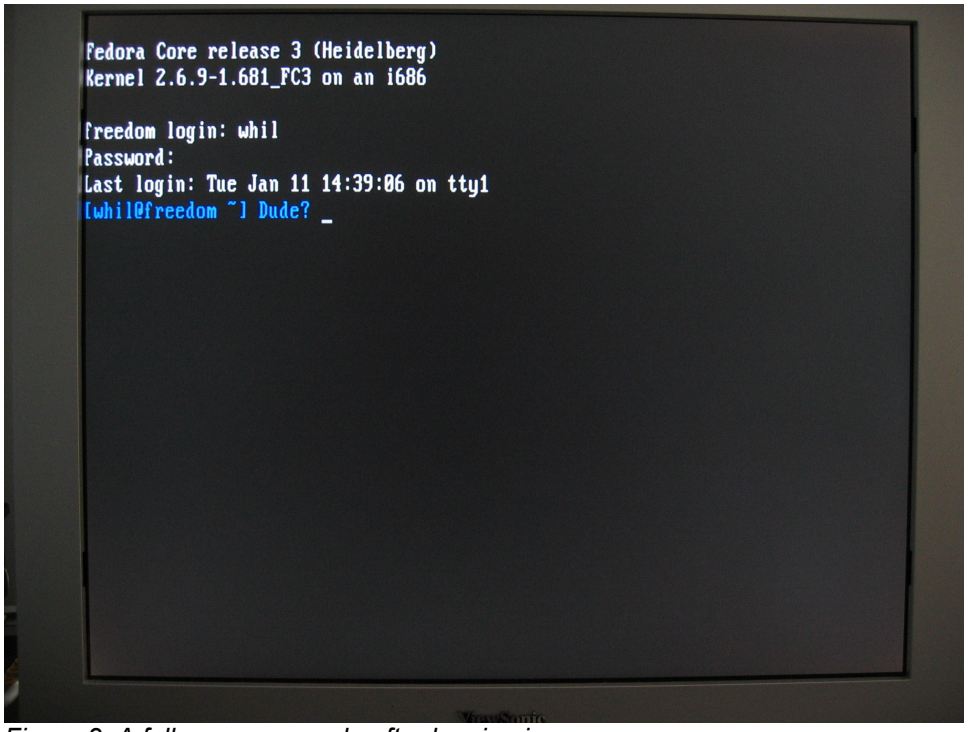

Figure 2. A full-screen console after logging in.

The standard Red Hat and Fedora Core configurations set up Linux with six separate consoles, available via Ctrl-Alt-F1 through Ctrl-Alt-F6.

Finally, once you've switched to a full screen console, you will eventually run into the situation where you want to switch back to the GUI session. Perhaps you're following along here, have tried switching to a full-screen console, and then realized you don't know how to get back to your GUI. Ctrl-Alt-F7 will do that for you.

6. Using PS and KILL to find and destroy processes

If the Ctrl-Alt-Backspace action as described previously didn't work, your next option will be to kill individual processes that are involved with the window and session managers. However, how do you find out what those processes are, and how do you kill a process once you've identified it?

Using PS

The PS command displays a list of the current processes running on the machine. For example:

```
# ps
PID    TTY      TIME  CMD
11048  pts/1  00:00:00  bash
11584  pts/1  00:00:00  ps
```

There are oodles of options that customize the display of what information is reported. Try "man ps" for details. For our purposes now, try ps aux to show you all processes owned by you or to list all processes when used together with the 'a' option. The listing can be rather long. A sample entry looks like this:

```
# ps aux
USER  PID  %CPU  %MEM  VSZ  RSS  TTY    STAT  START  TIME  COMMAND
root  11048 0.0 0.2 8804 2072 ? Ss Dec16  0:00  /sbin/cardmgr
whil  11433 0.0 0.1 4643 7288 ? S  10:12  0:00  /usr/bin/gnome-session
```

Including the 'f' modifier shows you the process hierarchy is displayed using ASCII art, like so:

```
# ps auxf
USER  PID  %CPU  %MEM  VSZ  RSS  TTY    STAT  START  TIME  COMMAND
root  1104 0.0 0.2 8804 2072 ?    Ss  Dec16  0:00  login -- whil
whil  2094                   tty1 Ss  Dec16  0:00   \_ -bash
whil  9111                   tty1 S+  10:12  0:00       \_ /bin/sh /usr/X11R6/bin/startx
whil  9200                   tty1 S+  10:12  0:00          \_xinit /etc/X11/xinit/xinitrc --
```

You'll see that the original login process was started on December 16, as was the bash shell. startx was started today at 10:12 (likely because I was testing the Ctrl-Alt-Backspace key combination and shut down X this morning.) xinitrc spawned off of startx, and so on.

Finding processes that you're interested in

Now that you know how to do general process listings, you'll see that there are a lot of processes running on your system. A lot. And it can be difficult to determine which ones are processes that you want to kill.

Unfortunately, there isn't a simple cookbook for doing this, because which window manager, display ,manager, graphical login screen you've got running on your system may well differ from what the guy next door has running on his. Thus, you'll need to do a bit of research.

Killing a process once you've found it

Supposing that you've identified a process that you want to kill, the next step is to actually do away with it, as they say in the movies, "with extreme prejudice". The KILL command combined with the process ID is the tool to use. Look through the process listing, find the process of interest, and then look under the "PID" ("Process ID") column to find the number of that process. Let's suppose that the PID is 12345. Issue the command

```
kill 12345
```

and press Enter at the command prompt. When you do another 'ps' listing, you'll see that the 12345 process is no longer running. Depending on what other processes lived under that process, you may find them to be gone as well.

Be careful when using the kill command; typing the wrong process number may end up making your system unstable, or crashing it altogether if you happen to have picked a bad choice.

According to the man page for kill, the "TERM" signal is sent to the specified process by default when no other signal is sent, and will kill the processes that do not catch it otherwise. If a process catches the TERM signal, you may need to use the "-9" parameter, since this signal can't be caught.

7. Your Second Recourse: the KILL command

If Ctrl-Alt-Backspace didn't work, killing the X server with kill -TERM or kill -9 in the console is your next best bet. This done like so:

```
kill <X_server_pid>
```

Or

```
kill -9 <X_server_pid>
```

as described in the previous section.

8. Your Third Recourse: Kill off Window then Session manager

If the X server won't die with either Ctrl-Alt-Backspace or an explicit kill command, you could try killing the window manager and then the session manager. However, the consensus was that if the X server hasn't died by now, the attempts to kill it have likely made it a zombie. Zombies are described in more detail in section 10.

One suggestion in this situation was to trying starting X all over again (using the startx command), but that should be viewed more as a short term solution, as there is probably something bad or unstable running around in the system by now.

9. Your Fourth Recourse: Use init

So by now we've determined that simply pounding on the X server and associated programs isn't going to do the trick. At this point, many folks, particularly experienced Windows users, would be inclined to reboot.

Suppose, though, that for some reason, you don't want to reboot if possible. You could try switching back to a text console in single user mode, and then starting back up to multi-user graphical mode. Here's how.

Still from console, switch to single user by typing

```
init 1
```

at the command prompt. This has the same effect as have a runlevel of '1' in your /etc/inittab during bootup. The init command kills all processes and the loads back up using the commands in rc.d/rc1. When the system is done, you'll be at a full screen single user text console. Log in, and then type

```
init 5
```

which will run the multi-user and GUI scripts in rc.d/rc5.

10. Your Last Recourse: Reboot

Something really bad has happened if you're reading this. Best bet is to reboot at this point.

11. About zombies

Dan Cody added these notes about zombie processes:

The reason you can't 'kill' a zombie (as Tom alluded to a bit) is because they're already 'dead', hence the term. Technically of course, a zombie is 'undead' in the classic sense, but that's neither here nor there. A zombie process (which shows as a Z in the STAT column when you run ps xa or top and is usually followed by <defunct>) has already terminated itself by giving an exit() call to the system or through some other sort of uncaught signal. For it to not show up in the process table, it's parent process must perform a wait() system call or something similar. For the parent to give the wait() call to the child though, it(the parent) needs to get some information about the exit status of the child, but the child can't because for one reason or another it's borked. Once the parent receive the information on the child's termination status, that child process no longer appears in the process table.

If any process dies before its children do, init inherits the children. When they die, init calls one of the wait() functions to retrieve the child's termination status, and the child disappears from the process table. If you continually restart an application

(like apache for instance) but there is an underlying problem that's causing the zombie processes to begin with, and yet you continue to 'kill' off the parent processes, they'll continue to get inherited by init. If you've ever seen a system with hundreds of zombie processes, this is usually why.

At any rate though, zombie's aren't always 'bad', but they can be indicators of bigger problems like buggy applications, or memory that's going bad. If left unchecked, zombie's will continue to take up space in the process table, and that will also cause problems sooner or later with other applications. Not that it's good to leave them hanging around or anything :)

To clear zombies from the process table, you do have to reboot, or wait for the parent process to retrieve the child's termination or exit() status.

If you're continually getting zombie processes, the best bet to find out why/how it's happening is to trace the process from it's beginning to undead end with 'strace'. Run 'strace ls' next time you've got a terminal open to see an example.

12. Where to go for more information

The primary resource are the help files for ps and kill. Type "info kill", "man kill", and "man ps".

13. About the author

Whil Hentzen started out life in the early '80's as a custom software developer using dBASE II (he still has the original 8 1/2 x 11 grey binder of documentation, much to the chagrin of his wife), and switched to FoxPro in 1990. Besides billing 15,000 hours in the 90's, he presented more than 70 papers at conferences throughout North America and Europe, edited FoxTalk, Pinnacle Publishing's high end technical journal for 7 years, hosted the Great Lakes Great Database Workshop since 1994. He's written 7 books and published 30 more on a variety of software development topics. He was a Microsoft Most Valuable Professional from 1995 through 2003 for his contributions to the FoxPro development community, and received the first Microsoft Lifetime Achievement Award for Visual FoxPro in 2001.

Whil began using Linux on the desktop when OpenOffice.org became a standard in the mainstream distributions, as it spelled potential for custom application development in the future, and has been a Linux user, developer, and evangelist ever since. His first book on Linux, Linux Transfer for Windows Power Users, was published in early 2004.

He is available for new and legacy Visual FoxPro application development as well as Web and desktop development on Linux.

14. A word from our sponsor

This free whitepaper is published and distributed by Hentzenwerke Publishing, Inc. We have the largest lists of “Moving to Linux”, OpenOffice.org, and Visual FoxPro books on the planet.

We also have oodles of free whitepapers on our website and more are being added regularly. Our Preferred Customer mailing list gets bi-monthly announcements of new whitepapers (and gets discounts on our books, first crack at special deals, and other stuff as we think of it.)

Click on “Your Account” at www.hentzenwerke.com to get on our Preferred Customer list.

If you found this whitepaper helpful, check out these Hentzenwerke Publishing books as well:

Linux Transfer for Windows® Network Admins:
A roadmap for building a Linux file and print server
Michael Jang

Linux Transfer for Windows® Power Users:
Getting started with Linux for the desktop

Whil Hentzen