Bugs in my program.

Harry Wang

Feb, 5\textsuperscript{th}, 2004
Goals

- How reliable are the UNIX/Windows utilities?
  - Look into other sources.

- Other interesting questions regarding UNIX/Windows system.
  - What is being done about bugs lurking in UNIX/Windows utilities?
  - Do people take bugs in UNIX/Windows utilities seriously?
  - What makes a program reliable?
papers 1990 - 1995 - 2000

- **An Empirical Study of the Reliability of UNIX Utilities.** Barton P. Miller, Lars Fredriksen, Bryan So. Communications of the Association for Computing Machinery, 33(12), 1990. (testing done on UNIX utilities only)

- **Fuzz Revisited: A Re-examination of the Reliability of UNIX Utilities and Services.** Barton P. Miller, David Koski, Cjin Pheow Lee, Vivekananda Maganty, Ravi Murthy, Ajitkumar Natarajan, Jeff Steidl, 1995, 1998 (tested more UNIX platforms tested, network services, X-window applications)

Many UNIX utilities are unreliable – crash or hang. (25%-33%)

Method: Automatic testing tool “fuzz” use random character inputs to “break the code”.

Problems:

- Few people report these bugs. (I.e. Expert users)
- Some common utilities ftp, csh, emacs, spell, telnet, vi are buggie.
- AIX UNIX - commercial version of UNIX performed no better than free versions of UNIX.
What to do from here?

- The authors mention their research “opened a door for much more in-depth research by others”. Has it happened?

- In 1995, do we have same number of bugs as in 1990?

- Are commercial version of UNIX better?
1995 paper

- Authors tested the **reliability** of a large collection of basic utility programs, X-Window applications and servers, and network services.

  - Tested 9 versions of the UNIX operating system, including seven commercial systems and the freely-available GNU utilities and Linux.
What systems were tested

- **From Sun Microsystems:**
  - SPARCstation 10/40 Running SunOS 4.1.3
  - SPARCstation 10/40 Running Solaris 2.3

- **From HP:**
  - HP 9000/705 Running HP-UX 9.01

- **From IBM:**
  - RS6000 Running AIX 3.2

- **From Silicon Graphics:**
  - Indy Running IRIX 5.1.1.2

- **From DEC:**
  - DECstation 3100 Running Ultrix v4.3a rev 146

- **From NEXT:**
  - Colorstation (MC68040) Running NEXTSTEP 3.2

- **GNU:**
  - Running SunOS 4.1.3 & NEXTSTEP 3.2

- **Linux:**
  - Cyrix i486 Running Slackware 2.1.0
1995 Result

- UNIX: Failure occurred in 40% (in the worst case) of the UNIX utilities. But GNU/LINUX utilities are much more reliable. (Testing tool: Fuzz)

- Network services: In 1990 the authors were able to crash ftpd and telnetd. But in the 1995 study they were not able to crash any of the services that were tested on any UNIX System. (Testing tool: Fuzz and portjig)

- X-Window applications: “Well more than half of the X-Window applications that were tested crash on random input data streams. More significant is that more than 25% of the applications crash given random, but legal X-event streams.” Failure is 58% at the worst case. (Testing tool: xwinjig)
### 1995 Result 2 (*)

<table>
<thead>
<tr>
<th>Utility</th>
<th>SunOS</th>
<th>HP-UX</th>
<th>AIX</th>
<th>Solaris</th>
<th>Irix</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>90</td>
<td>95</td>
<td>90</td>
<td>95</td>
<td>95</td>
</tr>
<tr>
<td># tested</td>
<td>77</td>
<td>80</td>
<td>72</td>
<td>74</td>
<td>70</td>
</tr>
<tr>
<td># crash/hang</td>
<td>22</td>
<td>18</td>
<td>24</td>
<td>13</td>
<td>16</td>
</tr>
<tr>
<td>%</td>
<td>29%</td>
<td>23%</td>
<td>33%</td>
<td>18%</td>
<td>23%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Utility</th>
<th>Ultrix</th>
<th>NEXT</th>
<th>GNU</th>
<th>Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>95</td>
<td>95</td>
<td>95</td>
<td>95</td>
</tr>
<tr>
<td># tested</td>
<td>80</td>
<td>75</td>
<td>47</td>
<td>55</td>
</tr>
<tr>
<td># crash/hang</td>
<td>17</td>
<td>32</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>%</td>
<td>21%</td>
<td>43%</td>
<td>6%</td>
<td>9%</td>
</tr>
</tbody>
</table>

* (Image taken from Power point presentation on Fuzz re-visited Re-Examination of Reliability of UNIX Utilities and Services)
1995 conclusion

- "The reliability of the basic utilities from GNU and Linux were noticeably better than those of the commercial systems."

- "all systems that we compared between 1990 and 1995 noticeably improved in reliability, but still had significant rates of failure."

- "We also tested how utility programs checked their return codes from the memory allocation library routines by simulating the unavailability of virtual memory. We could crash almost half of the programs that we tested in this way."
2000 paper – testing on Windows NT 5.0 (Windows 2000)

Applications tested:


- Acrobat Reader 4.0, Codewarrior Pro 3.3, Command AntiVirus 4.54, Eudora Pro 3.0.5, FrameMaker 5.5, Ghostscript 5.50, Ghostview 2.7, GNU Emacs 20.3.1, Java Workshop 2.0a, Netscape Communicator 4.7, Paint Shop Pro 5.03, Secure CRT 2.4, Solitaire 4.0, Telnet 5 for Windows, Visual C++ 6.0, Winamp 2.5c, WS_FTP LE 4.50.
Method 1: Give valid random mouse and keyboard input streams.

Result:
- Similar to the last two studies, large number of bugs are found in Windows NT utilities. 21% applications crashes and 24% hung.

Method 2: Give random Win32 messages.

Result:
- 100% applications failed (crashes or hung). The flaw is in the Win32 message interface itself - receiving a Win32 message and unsafely using a pointer or handle contained in the message.
Concern: Any of these events could be caused by a user!

After crash, no meaningful dialog message displayed. The error message containing hexadecimal memory dump address is confusing to average users.

In some cases, when UNIX utilities, the OS crashes. This did not happen in Windows NT.

Lack of source code make diagnose hard. (Only open-source projects GNU Emacs and Mozilla had source code.)

Venders forbidden researchers to report bugs! (I.e. certain database venders)
What is being done about these bugs lurking in UNIX utilities?

- Researches are being conducted. New questions are raised.
  - Symeon (simos) Xenitellis at Univ. of London is studying vulnerabilities in event-driven systems specifically in Windows 95, 98, NT, XP (Prevent malicious attacks) ([http://www.isg.rhul.ac.uk/~simos/event_demo/](http://www.isg.rhul.ac.uk/~simos/event_demo/))

- In 2001, Brian L. Bowers wrote a paper *An Inquiry into the Stability and Reliability of UNIX Utilities* to concentrate on determining the reliability of modern day utilities on GNU/Linux and Solaris machines (Red Hat 6.2 and SunOS 5.7). They tested tcsh, csh, sh, vim, grep, latex, make, more, tee, yacc, lint, etc. Results: using fuzz, worst case UNIX(11%), GNU/Linux systems are most reliable(4%). (good read)
What is being done about these bugs lurking in UNIX utilities? (2)

- Ben Woodward was inspired to create an alternate version of fuzz, for Linux, and release it under the GPL. (In active development; didn’t compile on Solaris.) (http://fuzz.sourceforge.net)

- “The Bulletproof Penguin” website maintained by Scott Maxwell is focused on bringing about the fixes of all bugs found in the 1995 fuzz paper and elsewhere. Scott Maxwell will send patches to all who want them. (“fuzz bugs”)
  (http://home.pacbell.net/s-max/scott/bulletproof-penguin.html)

- Future work: test network related daemon software - BIND, sendmail, ftpd, and fingerd.
Do people take bugs in UNIX utilities seriously?

- Yes, but most problems are still caused by the simplest mistakes - arrays and pointers, not checking return codes, assuming correct inputs (too much trust in other segment of code), and stop using `gets()` - using `fgets()` instead to limit the length of the input line, etc.

- Yes, but many old bugs have not been fixed. And new bugs keeps on coming...

- Microsoft releases patches often. But who really cares? As an expert user, when did you last report bugs to Microsoft, an UNIX administrator...?

- “The Bulletproof Penguin” is taking bugs in GNU/LINUX utilities very seriously.

Open source projects better than commercial projects?

“Why a globally scattered group of programmers, with no formal testing support or software engineering standards can produce code that is more reliable (at least, by our measure) than commercially produced code?”

- According to Miller, better programmers with higher motivation to fix problems in open source projects makes create reliable programs. “The people who write and support the GNU utilities are typically skilled at their profession and are in it for the fun and intellectual challenge.”
- Better communication between authors and users. (direct email)
- Ubiquitous source code makes it easy to fix problems.
- Sense of accomplishment.
- Corporate bureaucracy.
- What do you think?

+ redhat. VS.
Want to run “fuzz” yourself?


- Scott Maxwell is leading an effort to eliminate "fuzz bugs" from GNU software.
Acknowledgments

(*)Power point presentation on Fuzz re-visited Re-Examination of Reliability of UNIX Utilities and Services

Cormac Flanagan

Google.com

* (Image taken from http://www.google.com)
Vulnerabilities in event-driven systems

“In event-driven systems it is common to be able to send events (=messages) from unprivileged users to privileged users (guest - Administrator). In Windows 2000, an unprivileged process (example: Trojan horse) can enumerate all windows and identify the important ones for the title bar and so on. Then, it can send events to them with PostMessage(). There is currently no protection as to who has sent the message. One can use it to send custom events but the most interesting aspect is the sending of legitimate messages to instruct the victim to do things you want it.”

Symeon (simos) Xenitellis
Additional points

Fuzz testing only reveals a portion of bugs in UNIX/Windows utilities. Chances are many more bugs still remain....

Additional points

- http://www.gnu.org/software/reliability.html “Free Software is More Reliable!”

- It maybe unfair to say Windows is less reliable than open source projects. After all, windows applications are much bigger and nicer to operate. Windows applications have more user-friendly GUI than tends to cause more bugs too.