LASIGE
Large-Scale Informatics Systems Laboratory

Finding Vulnerabilities in Software Ported from 32 to 64-bit CPUs

Ibéria Medeiros and Miguel Correia
University of Lisboa, Fac. Ciências, LASIGE

DSN, July 2009

The problem

- CPUs are moving from 32 to 64 bits
- In C in 32-bit CPUs we had:

  sizeof(int) = sizeof(long) = sizeof(pointer)

- No longer true for 64-bit CPUs
- When software is ported from 32 to 64 bits without care, bugs and vulnerabilities appear

Data models

Type	ILP32	LP64	LLP64	ILP64
	ANSI	*Unixes*	*Microsoft*	*Cray, etc.*
int	32	32	32	64
long	32	64	32	64
pointer	32	64	64	64

32 bits 64 bits

- Programmers with ILP32 think…
  - *sizeof(int) = sizeof(long) = sizeof(pointer)* so why care when moving data?
  - and by default functions return *int* …
- What if program (badly) ported from ILP32 to LP64?
  - What if there are assignations *int = long* or *int = pointer* ?
  - *Truncation, a kind of integer overflow!*

3

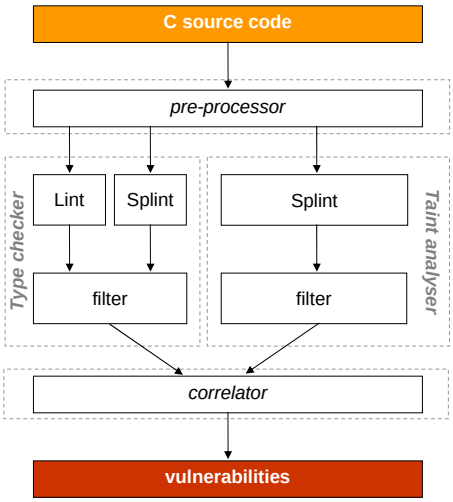
Finding these vulnerabilities

- Approach: *static code analysis*
  - meaning analysis of source code without execution
- We need two kinds of analysis:
  - Type checking → for finding the integer manipulation bugs
  - Taint analysis → for finding if data coming from input reaches dangerous places
- We didn't build the tool from scratch
  - We used 2 existing tools: Lint and Splint

4

The tool – DEEEP

- Detector of integEr vulnerabilitiEs in softwarE Portability



5

Experimental results

Application	Version	Bugs (type checker)	Vulnerabilities (correlator)	False positives (human insp.)	Files	Lines of code	Analysis time
wu-ftpd	2.6.2	217	0	-	50	22.629	25 seg
vsftpd	2.0.5	91	0	-	34	12.376	21 seg
sendmail	8.14.1	1.132	3	3	160	112.700	1:52 min
samba	3.0.26a	21.566	0	-	651	494.688	23:11 min
proftpd	1.3.0a	409	0	-	95	87.868	1:30 min
lighttpd	1.4.18	886	0	-	93	52.134	2:00 min
inetutils	1.5	980	0	-	175	79.793	1:16 min
dovecot	1.0.5	1.984	0	-	359	111.026	5:58 min
bind	9.4.1	1.298	0	-	604	323.860	4:24 min
asterisk	1.4.18	11.906	2	2	414	333.997	10:28 min
antiword	0.37	255	1	1	67	30.864	1:08 min
aircrack	0.9.3	353	4	4	14	21.147	0:35 seg
lsat	0.9.6	13	1	1	36	6.923	0:18 seg
ipac-ng	1.31	206	1	1	28	19.928	0:29 seg
rodmap	1.1.0	719	4	4	150	69.929	3:28 min
pen	0.17.2	95	1	1	5	3.772	0:21 seg
atop	1.22	169	3	3	14	12.030	0:32 seg
clamav	0.60	495	3	3	50	19.873	1:10 min
openldap	2.1	310	2	2	452	198.626	9:59 min
openafs	1.4.6	8.290	0	-	1221	689.920	15:25 min
wine	0.9.52	715	0	-	2197	1.833.251	1:57:38 h
TOTAL	**21**	**52089**	**25**	**25**	**6869**	**4537334**	

6

Conclusions

- A subtle problem caused by porting code from 32 to 64-bit CPUs
  - People are arguing that it is a big problem
- We built a tool to find these vulnerabilities
- We wrote synthetic code with such vulnerabilities – DEEEP detected all of them
- …but detected no vulnerabilities in 21 open source applications (> 4.5 Million LOCs)
- A surprising negative result

LASIGE
Large-Scale Informatics Systems Laboratory

Thank you. Questions?

http://deeep.homeunix.org