Analysis of Code Protection Technologies in ELF binaries in Major Linux Distributions and Generations

Shota SUGAWARA‡, Ryohei WATANABE‡, Shuta KONDO‡, Masahiro YOKOYAMA‡
Jiai NAKAMURA†, Kuniyasu SUZAKI*, Takamichi SAITO†

Graduate School of Meiji University‡ Meiji University† National Institute of Advanced Industrial Science and Technology*

Introduction

- Some code protection technologies are implemented on major compilers, and they are widespread
- But it is unknown whether they are applied and worked correctly

We surveyed
- the application situations of 4 code protection technologies on GCC (RELRO, SSP, PIE, Automatic Fortification)
- how the situations have been changed

by analyzing binaries on each 3 versions of 3 major linux distributions

Our Analysis

Target distributions

All distributions are 32bit

Distribution	Version		
CentOS (affiliated with Red Hat)	**5.0** release date: 04/12/2007 end of support: 03/31/2017	**6.0** release date: 11/09/2007 end of support: 11/30/2020	**7.3** release date: 12/12/2016 end of support: 06/30/2024
openSUSE (affiliated with Slackware)	**12.1** release date: 11/16/2011 end of support: 05/06/2013	**13.1** release date: 11/19/2013 end of support: 02/03/2016	**13.2** release date: 11/04/2014 end of support: 01/17/2017
Ubuntu (affiliated with Debian)	**10.04** release date: 08/17/2010 end of support: 05/06/2013	**12.04** release date: 04/26/2012 end of support: 04/28/2017	**14.04** release date: 04/17/2014 end of support: 04/??/2019

Target binaries

All binaries in the directories which set to root user's PATH environment variable by default

Total number of target binaries		
CentOS 5.0	CentOS 6.0	CentOS 7.3
1,700	1,432	1,579
openSUSE 12.1	openSUSE 13.1	openSUSE 13.2
2,033	2,169	2,194
Ubuntu 10.04	Ubuntu 12.04	Ubuntu 14.04
1,131	1,094	1,160

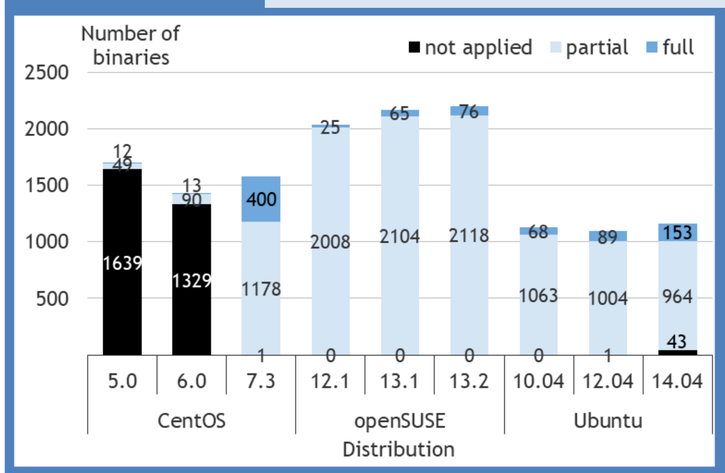
Analysis way

We checked particular headers, segments, sections, and symbols in each binary to find whether 4 code protection technologies are applied.

Result1: application situations of 4 code protection technologies

RELRO

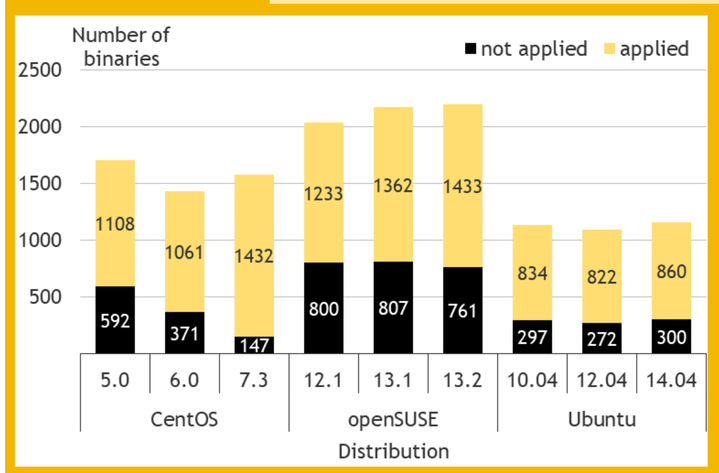
- the mechanism which sets particular sections read-only
- Full RELRO sets all possible sections read-only
- Partial RELRO does not set a part of the GOT section read-only



- Any RELRO was not applied to over 90% of binaries in CentOS 5.0 and CentOS 6.0
- Partial RELRO was applied to at least 75% of binaries in the distributions except for CentOS 5.0 and CentOS 6.0
- Full RELRO was applied to 25% of binaries in CentOS 7.3, and this is the highest proportion

SSP

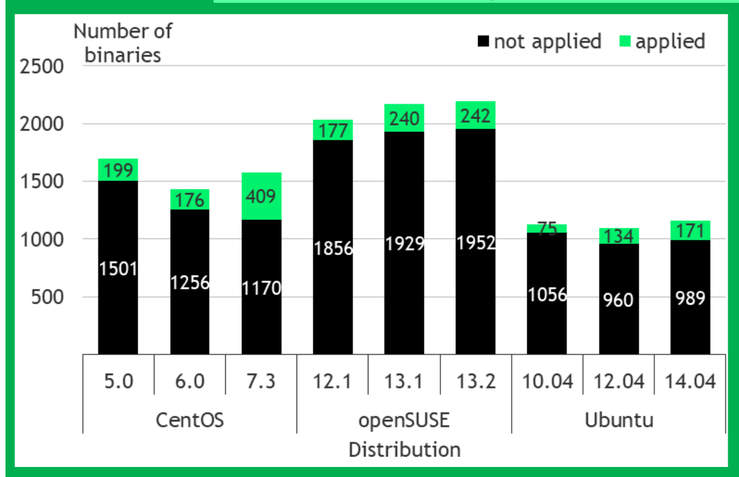
- the mechanism which sets a canary value on the saved ebp in the stack frame
- it detects buffer overflow if the canary value was overwritten



- In any distributions, the proportion of binaries which were applied SSP was high (at least 61% in openSUSE 12.1)
- CentOS 7.3 has the highest proportion (91%)

PIE

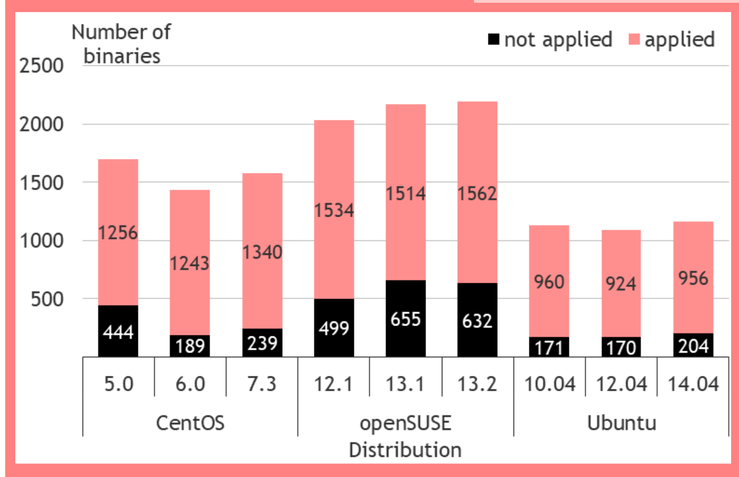
- the mechanism which generates position-independent executables
- If PIE and ASLR are applied to a binary, the base address of text area in the binary can be also randomized.



- In any distributions, the proportion of binaries which were applied PIE was low (at most 26% in CentOS 7.3)
- Especially, it was applied to only 7% of binaries in Ubuntu 10.04, and this is the lowest proportion
- We think this is because PIE does not work effectively and causes high overhead in 32bit environment

Automatic Fortification

the mechanism which replaces library functions that can cause buffer overflow to safe ones



- In any distributions, the proportion of binaries which were applied Automatic Fortification was high (at least 70% in openSUSE 13.1)
- CentOS 6.0 has the highest proportion (87%)

Result2: transitions of applications

We surveyed how the applications have been changed as each distribution's version upgrade

Distribution	Version upgrade	Changes: Number of Strengthened / Weakened binaries				Number of common binaries
		RELRO	SSP	PIE	Automatic Fortification	
CentOS	5.0 => 6.0	25 / 8	59 / 21	18 / 12	64 / 14	957
	6.0 => 7.3	977 / 0	178 / 5	127 / 0	12 / 16	1,004
openSUSE	12.1 => 13.1	28 / 0	74 / 14	45 / 0	7 / 14	1,623
	13.1 => 13.2	8 / 2	24 / 9	0 / 3	14 / 8	2,052
Ubuntu	10.04 => 12.04	36 / 25	23 / 10	38 / 0	7 / 21	965
	12.04 => 14.04	54 / 46	4 / 11	32 / 7	20 / 40	1,006

- strengthen means the binary changes not applied => applied, or partial => full
- weaken means the binary changes applied => not applied, or full => partial

Weakened binary example (as version upgrade Ubuntu12.04 => Ubuntu14.04)

RELRO & Automatic Fortification
- RELRO and Automatic Fortification were weakened on same 33 binaries
- All of these binaries were related to X Window System

SSP 9 of 11 binaries were set SSP compile option in its source, but the generated binaries were disabled SSP.

➡ the code protection technologies are not always applied while the binaries were built with their compile option.

PIE
- PIE were weakened on 7 binaries
- All of these binaries were related to dbus
- According to the developers, they did not apply PIE because PIE disturbed normal execution

Conclusion

- The application situations are different on each distribution
- There are cases code protection technologies were disabled on later version though they had been enabled on previous version
- The code protection technologies are not always applied while the binaries were built with their compile option

Future Works

We will
- analyze the binaries in 64bit Linux distributions
- clarify the difference of these code protection technologies' situations between 32bit and 64bit