Metody przeciwdziałania przepełnieniu bufora

Bogdan Yakovenko

Metody przeciwdziałania

1. PaX dla Linuxa
2. DEP and ASLR for Windows
3. Metody ktore oferują kompilatory

Co to jest PaX?

- Patch dla jądra Linuxa
- Zestaw metod chroniących program od wykorzystania błędów oprogramowania w celach dostępu do pamięci procesu

Idea PaX

- PaX stosuje techniki które przeszkadzają sensownie wykorzystywać błędy oprogramowania
- PaX NIE stosuje żadnych technik żeby wykryć te błędy

Typy ataków na system

- Wprowadzenie/wywołanie dowolnego kodu
- Wywołanie istniejącego kodu w niewłaściwej kolejności
- Wywołanie isniejącego kodu we właściwej kolejności ale z dowolnymi danymi

Typy ataków na system

- Wprowadzenie/wywołanie dowolnego kodu
- Wywołanie istniejącego kodu w niewłaściwej kolejności
- Wywołanie isniejącego kodu we właściwej kolejności ale z dowolnymi danymi
- DoS – (**denial-of-service**) uniemożliwia działanie

Przepełnienie bufora

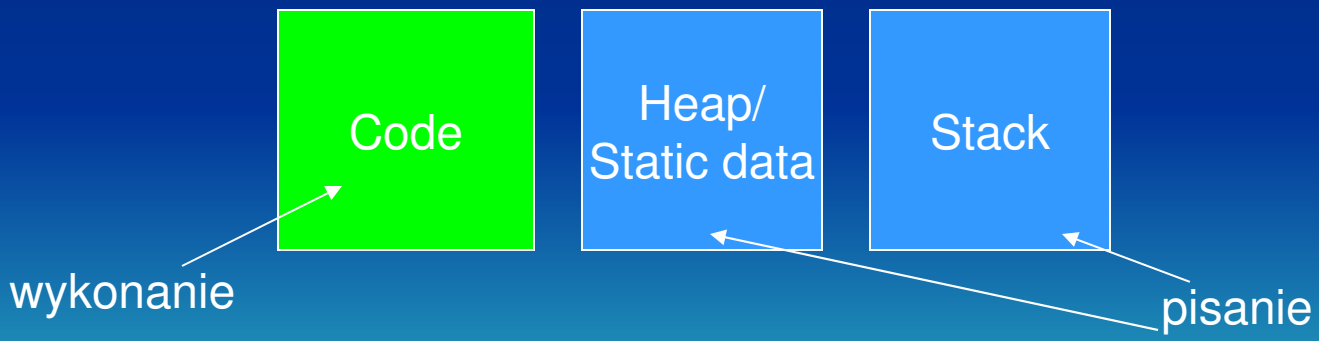
- Stack-based
  - Przepisywanie zmiennych lokalnych, które są blisko bufora dla własnych korzyści
  - Przepisywanie adresu powrotnego
    - Shellcode
    - Return-to-libc
- Heap-based i BSS-based
  - Przepisywanie zmiennych obok
  - Przepisywanie adresu funkcji

NOEXEC

- Cel: nie dopuścić do wprowadzenia i wykonania kodu w przestrzeni adresowej procesu
- Idea: zabronić tworzenia stron, do których można zapisać dane i wywołać kod.

NOEXEC

- Cel: nie dopuścić do wprowadzenia i wykonania kodu w przestrzeni adresowej procesu
- Idea: zabronić tworzenia stron, do których można zapisać dane i wywołać kod.



NOEXEC (features)

1. Realizować semantykę stron wykonywalnych na zwykłych stronach pamięci.
2. Jądro musi wykorzystowywać wprowadzonę semantykę
3. Zabrania stworzenia stron, do których można zapisać dane i wywołać kod.

Jak zaznaczyć stronę pamięci jako niewykonywalną?

- Wsparcie sprzętowe – NX bit (architektury: AMD64, SPARC, ALPHA, i t. d.)
- Emulacja (architektura IA32) *SEGMEXEC, PAGEEXEC*

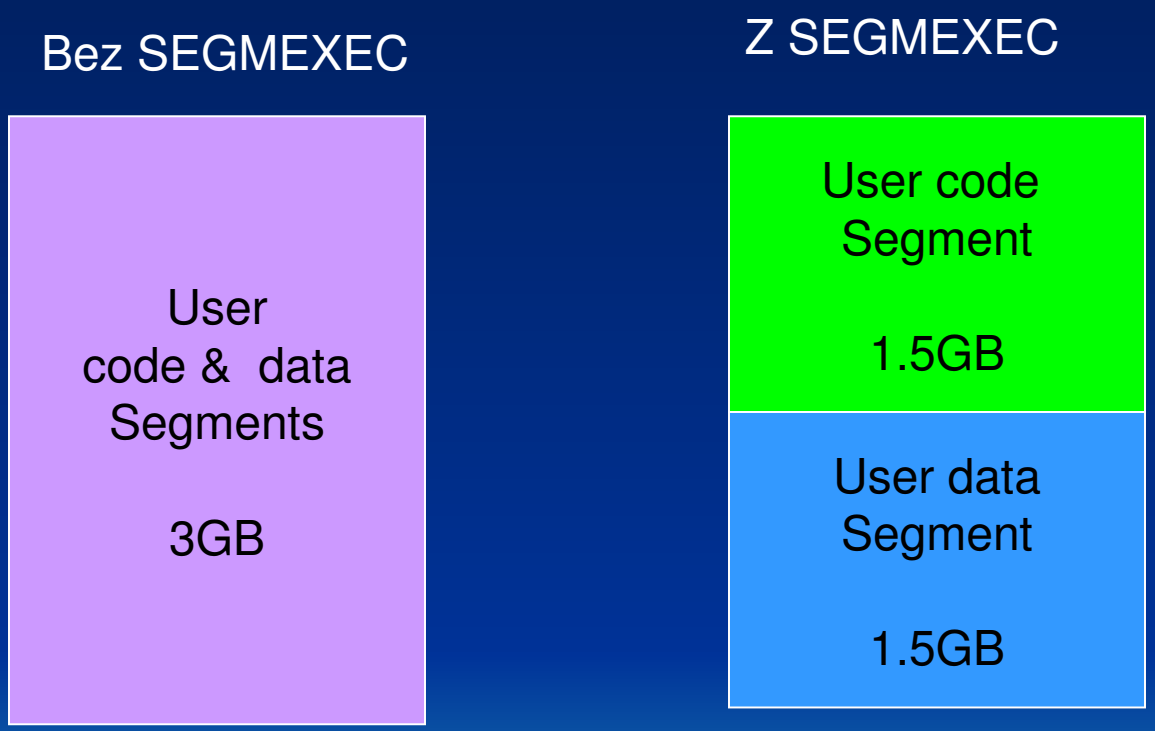
SEGMEXEC

- Jest to implementacja niewykonywalnych stron za pomocą logiki segmentacji IA-32 procesorów.

SEGMEXEC c.d.

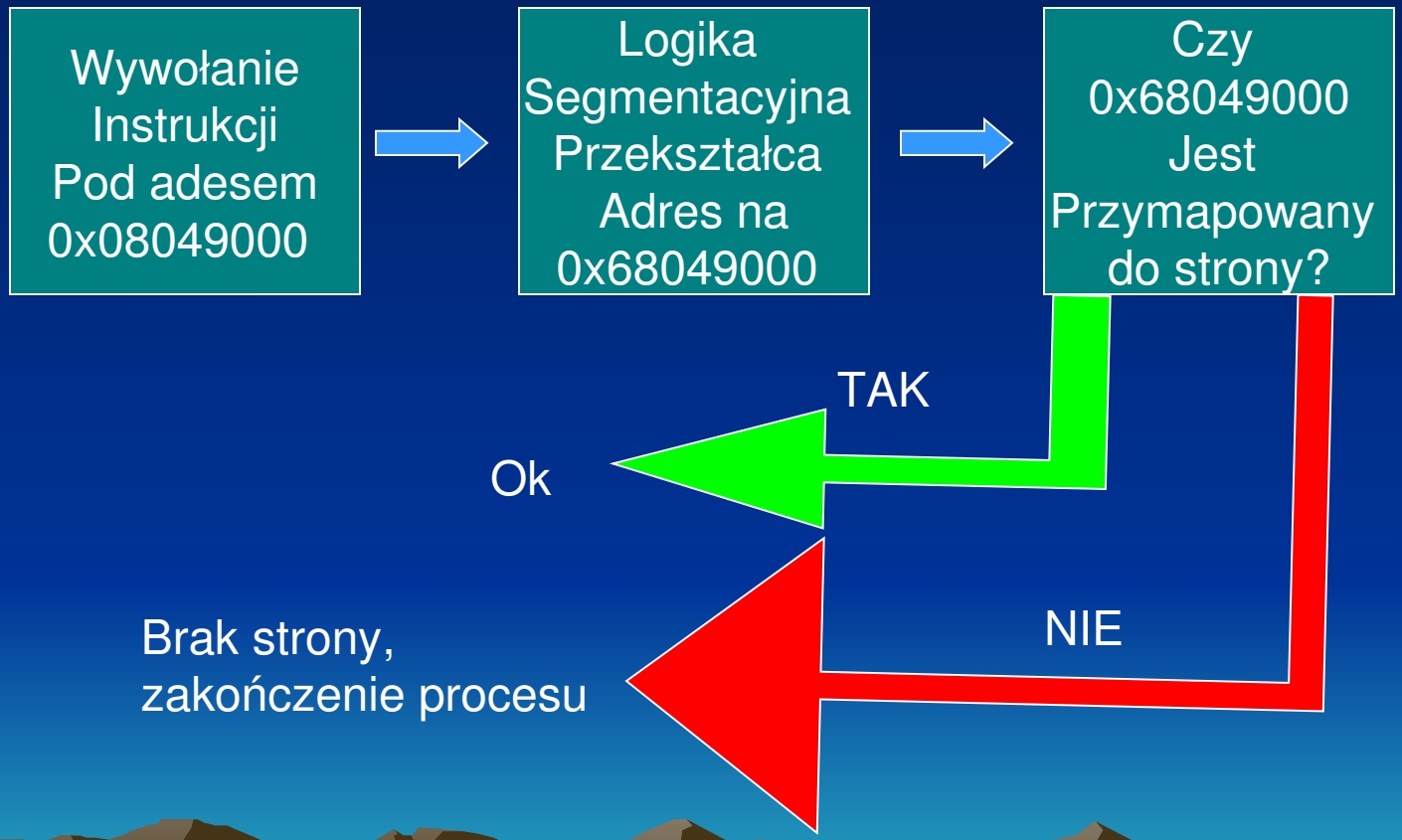
- Logika segmentacyjna jest bardzo prosta
  - Data segment
  - Code segment
- PaX dzieli przestrzeń adresowa na dwie równe części: dolna połowa jest dla danych, górna – dla kodu.
- Segmentacja jest „oknem” do przestrzeni adresowej

SEGMEXEC c.d.


Bez SEGMEXEC
User
code & data
Segments
3GB
Z SEGMEXEC
User code
Segment
1.5GB
User data
Segment
1.5GB

SEGMEXEC c.d.

- Każda wykonywalna strona z dolnej połowy jest odbita w górnej połowie. (Vitrual Memory Address Mirroring)
- Wywołanie instrukcji z adresów, które są w data segment i nie mają żadnego kodu znajdującego sie w odbitym adresie, powoduje brak strony. PaX przychwyci ten brak strony i zabije zadanie


Wywołanie Instrukcji Pod adesem 0x08049000
Logika Segmentacyjna Przekształca Adres na 0x68049000
Czy 0x68049000 Jest Przymapowany do strony?
TAK
Ok
NIE
Brak strony, zakończenie procesu

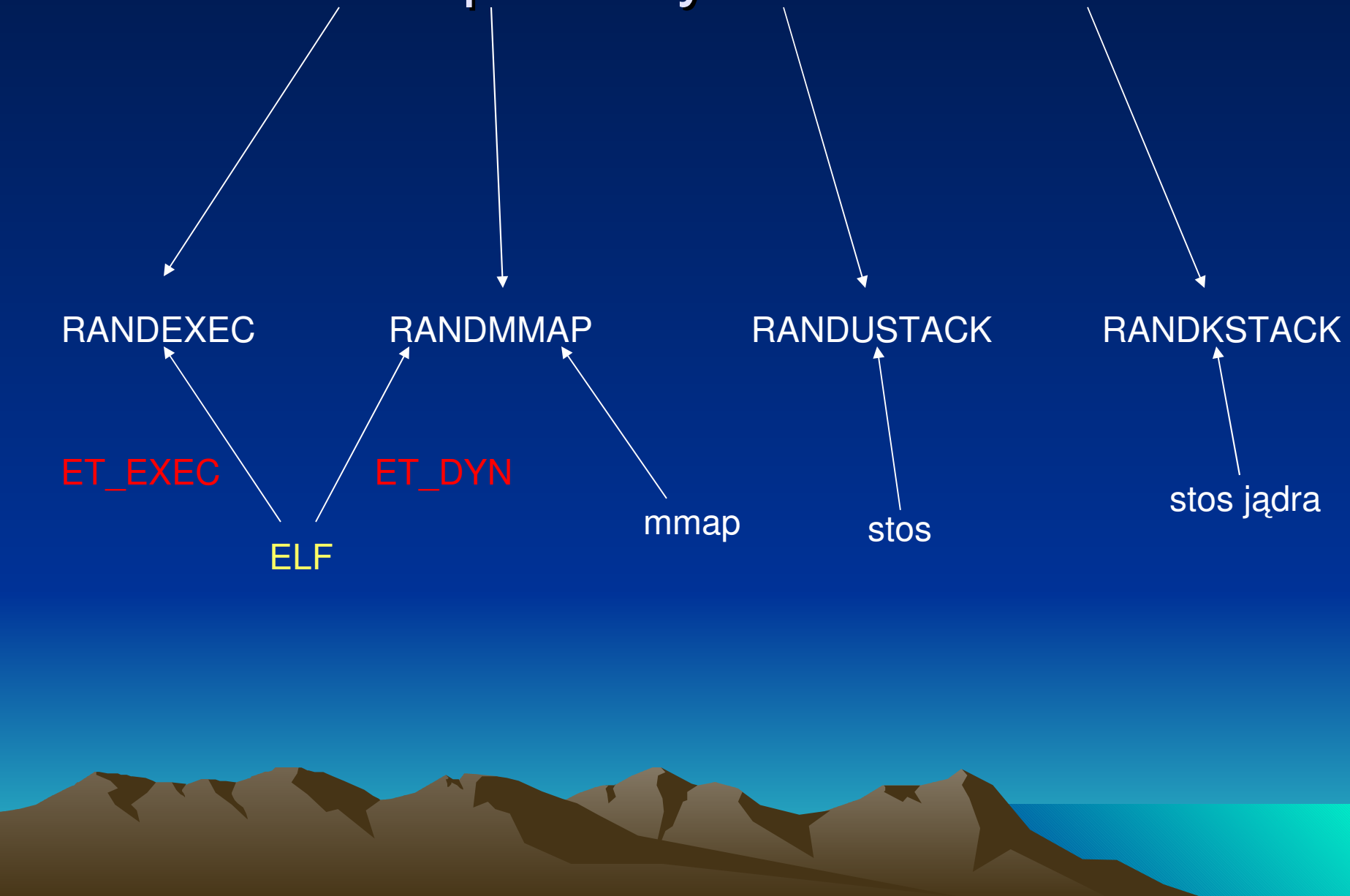
MPROTECT

- Cel: nie dopuścić do wprowadzenia nowego wykonywalnego kodu do przestrzeni adresowej procesu
- Idea: ograniczenie możliwości funkcji systemowych: mmap() i mprotect().

ASLR (Address space layout randomization)

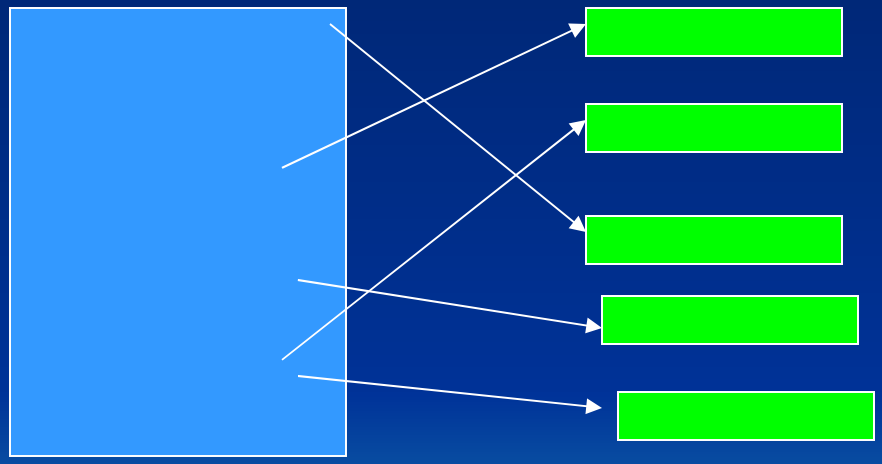
- Wprowadzenie losowości w rozłożeniu obiektów w przestrzeni adresowej.
- Technika przeciwko return-to-libc atak.

Address space layout randomization



RANDEXEC

- Randomizacja adresów mapowań plików ET_EXEC



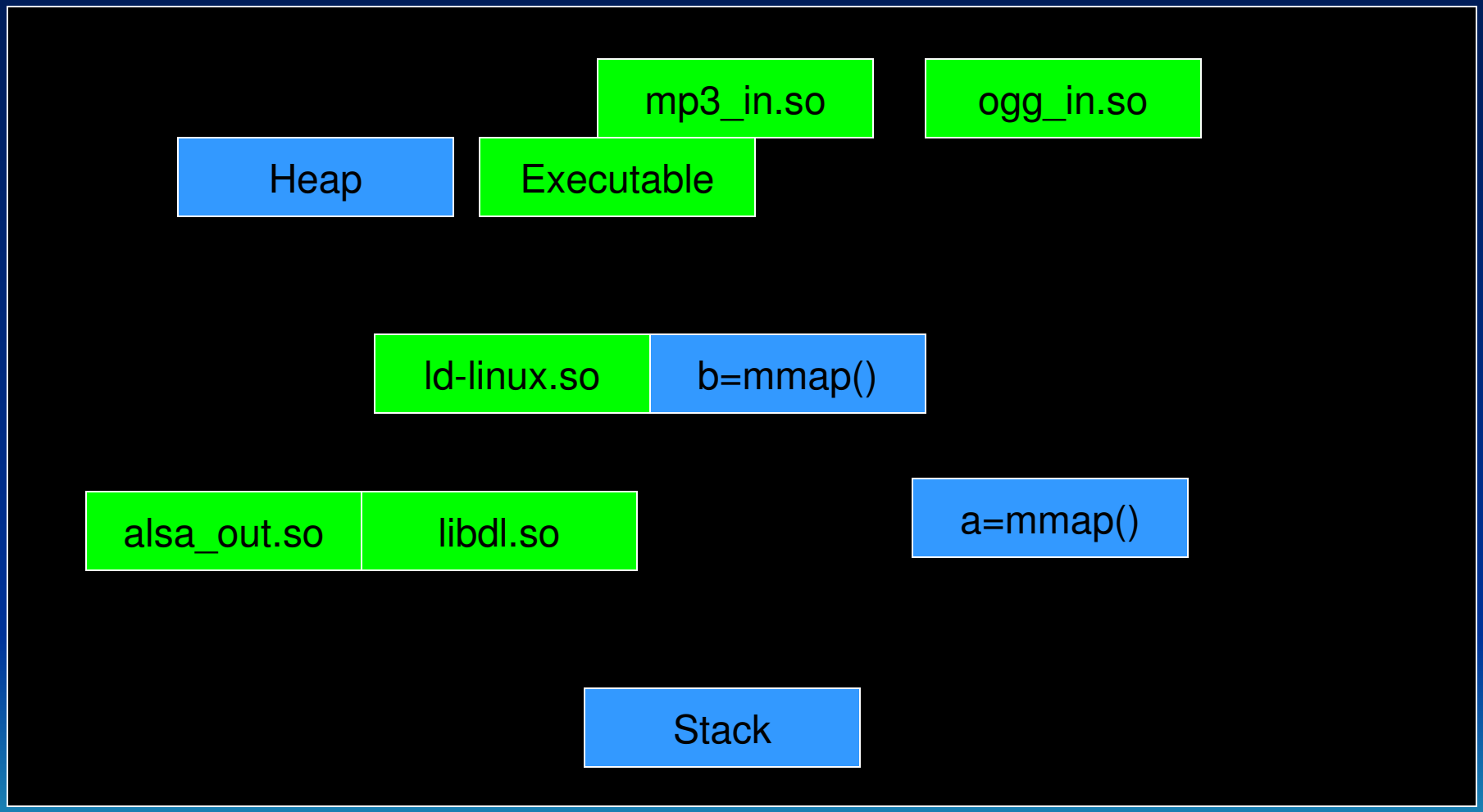
Oryginalne mapowanie
(niewykonywalne)

Losowe mapowanie
(wykonywalne)

ASLR c.d.

- RANDMMAP – Randomizacja obszarów pamięci obsługiwanych przez do_mmap()
- RANDUSTACK – Randomizacja adresów stosu wykonana podczas tworzenia proces
- RANDKSTACK Randomizacja adresów stosu jądra dla procesów (przed każdym powrotem z wywołania funkcji systemowej)

NOEXEC + ASLR


mp3_in.so
ogg_in.so
Heap
Executable
ld-linux.so
b=mmap()
alsa_out.so
libdl.so
a=mmap()
Stack

Windows

- Data Execution Prevention
  - Hardware DEP (NX byte)
  - Software DEP

  Powoduje problemy dla niektórych programów
- ASLR
  - Działa w Windows Vista
  - Jest podobny do wersji PaXa.

GCC Stack-Smashing Protector (ProPolice)

- Rozszerzenie GCC dla przeciwdziałania błędu przepełnienia stosu.
- Automatycznie dodaje chroniący kod do programu kompilowanego

GCC Stack-Smashing Protector (ProPolice) c.d.

- Dodatkowa zmienna choniąca kod przeciw zmian adresu powrotnego
- deklaracja lokalna int guard;
- Na wejściu guard = guard_value;
- Na wyjściu

```
if (guard != guard_value) {
        /* output error log */
        /* halt execution */
}
```

- Zmienne lokalne

  są za tablicami