gynvael.coldwind

IT Security

vs

GameDev

IGK'8 2011, Siedlce / Poland

gynvael.coldwind

/usr/bin/**whoami**

http://gynvael.coldwind.pl/

Plan

Część I

Różne definicje bezpieczeństwa

Część II

GameDev vs hacking

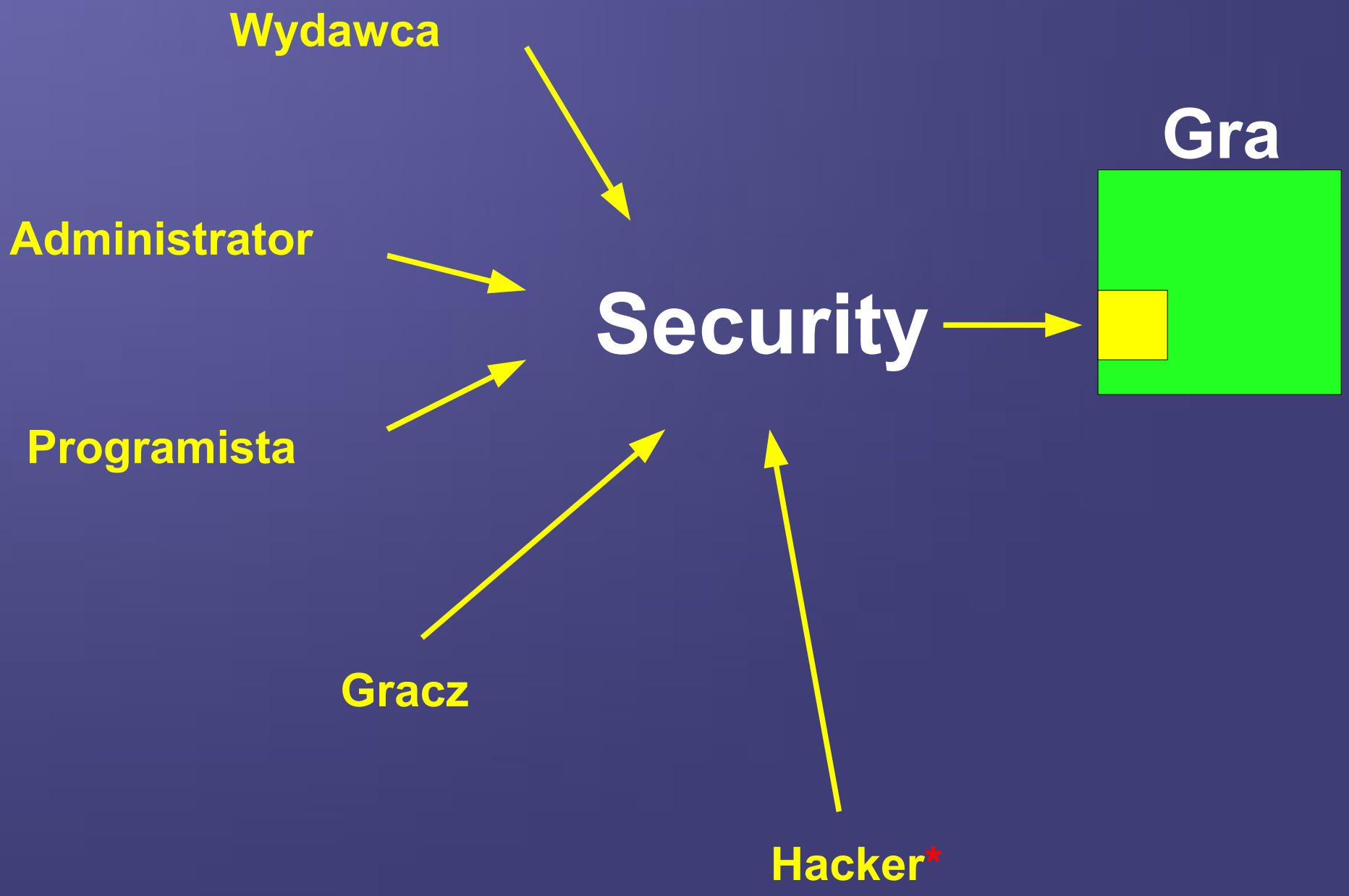
o czym nie będzie...

Część I

Różne definicje bezpieczeństwa

“Punkt widzenia zależy od punktu siedzenia”

(autor nieznany)


Wydawca
Administrator
Programista
Gracz
Hacker*
Security
Gra

Wydawca a security

J. Hutsch
http://commons.wikimedia.org/wiki/File:Piratenfahne_0002.jpg

DRM

Digital rights management *(DRM, pol. cyfrowe zarządzanie prawami)*
- oparty o mechanizmy kryptograficzne lub inne metody ukrywania treści
system zabezpieczeń mający przeciwdziałać używaniu danych
w formacie elektronicznym w sposób sprzeczny z wolą ich ***wydawcy****.*
(źródło: Wikipedia)

Niepoprawny optymista o DRM:
“DRM zabezbiecza naszą grę przed piratami”

YEAH...
RIGHT :)

Realista o DRM:

“DRM odsuwa w czasie 'piracenie' naszej gry”

W3, PS3 i Linux, next slide

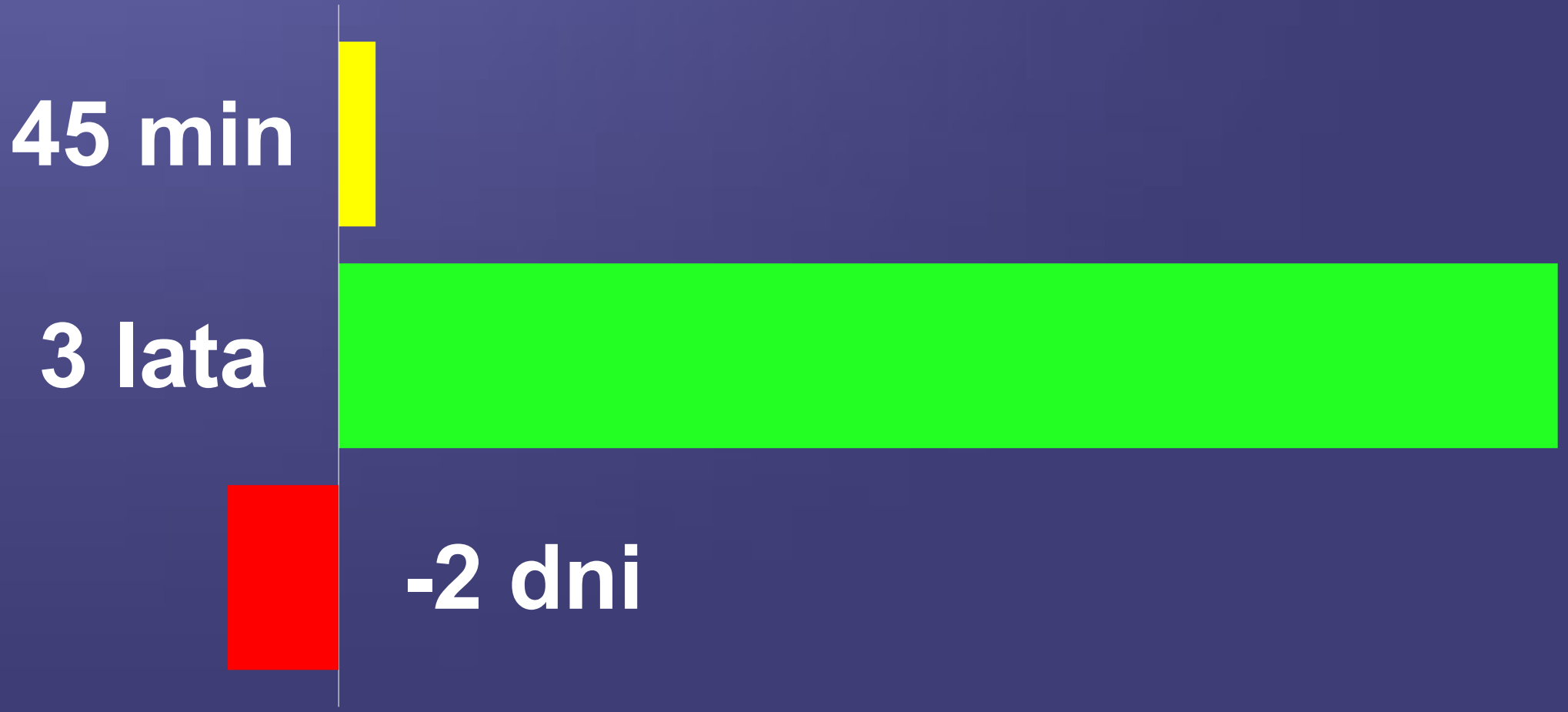
Realista o DRM:

“DRM odsuwa w czasie 'piracenie' naszej gry”

W3, PS3 i Linux, next slide

Realista o DRM:

“DRM odsuwa w czasie 'piracenie' naszej gry”



W3, PS3 i Linux, next slide

DRM, your doing it wrong!

Dygresja

Użytkownik (licencjonowany) o DRM:
“DRM mnie wkurza”

3 instalacje

2016 z 2216 * of 5

class action vs EA rootkit

most pirated game of 2008
-2 DAY

“Wszyscy jesteście złodziejami”
?id=299

Użytkownik (nielicencjonowany) o DRM:
“crack.exe<enter> DRM? Hmm? What's that?”

single player&connection

pad serwera
=
gra w menu

emu

“debuger w tle”

Admin a security

Cheat'y:

wallhack, maphack, teleport hack (C2),
aimbot, etc...

vs

anty-cheety
(pasywne / detekcja / prewencja)

Privacy & Accounts:

“*dobrze by było, gdyby nikt nam nie ukradł bazy danych*”

Gracz a security

Cheat & Accounts.. I trojany?

Pod koniec lipca do komendy policji w Olsztynie zgłosił się 27-letni mężczyzna. Twierdził on, że został okradziony. Ktoś zabrał mu magiczne buty, tarczę, naszyjnik, dwie bransoletki i hełm.

Swoje straty wycenił wstępnie na 900 zł. Te przedmioty jednak nigdy nie istniały w realnym świecie. To wyposażenie, jakie posiadał awatar 27-latka w popularnej grze internetowej Tibia.

źródło:
http://giernik.pl/policja-ukradli...
Michał Kowal

Wszystko to co chcą poprzednicy

+

Bezpieczeństwo implementacji

Hacker* a security

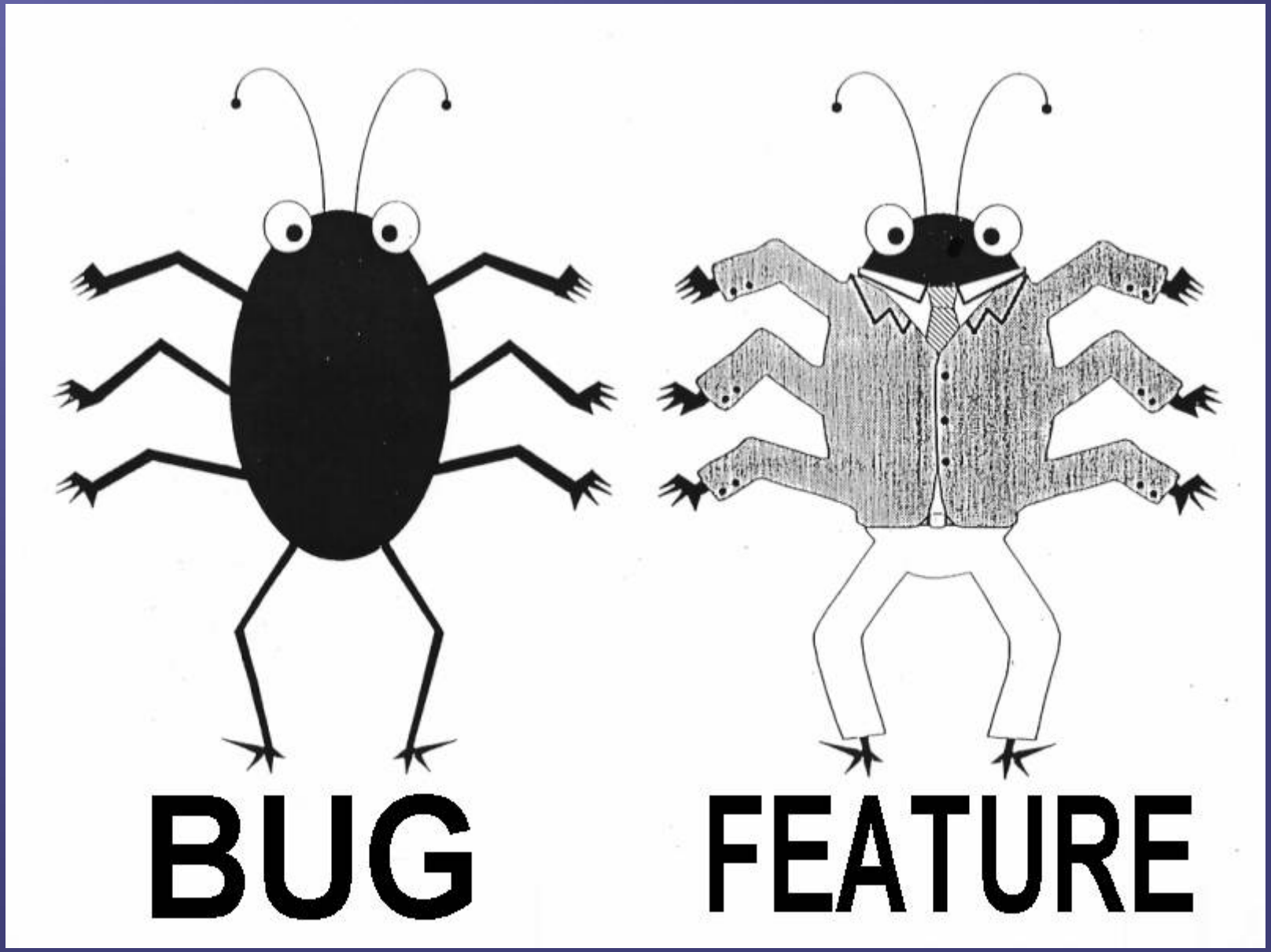
vs

Bezpieczeństwo implementacji

Część II

GameDev vs hacking

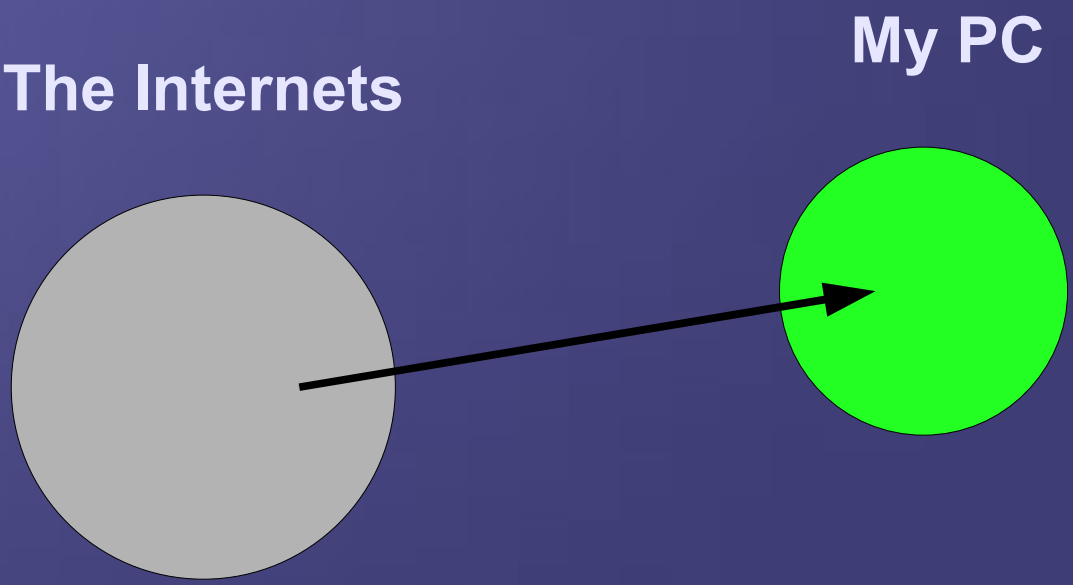
Bug vs ~~Feature~~ Vulnerability



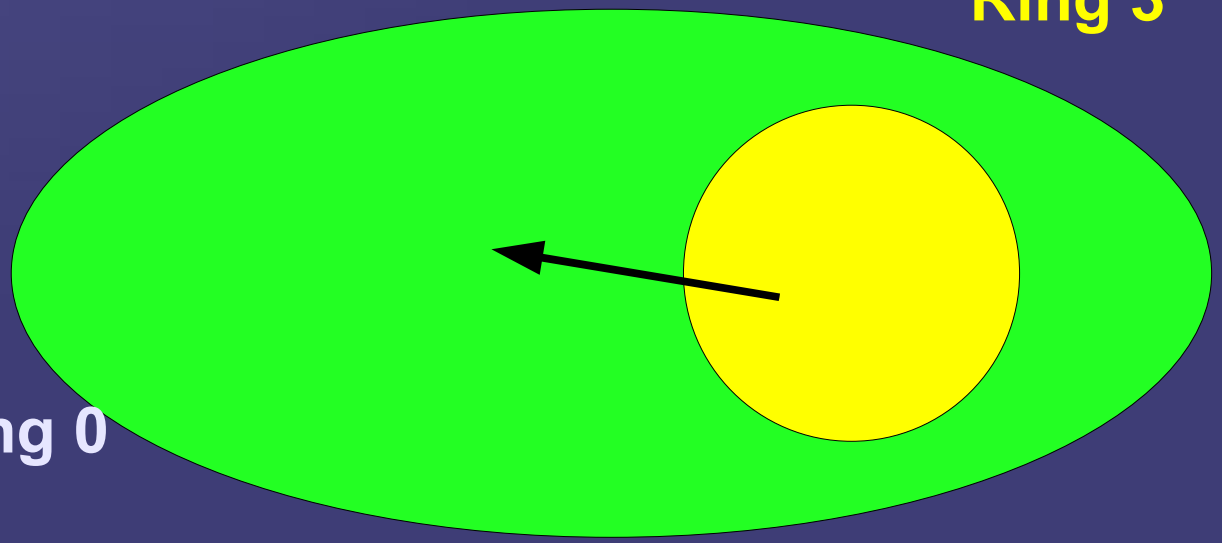
Bug vs Vulnerability

zwiększenie praw / przełamanie separacji

Wersja A:



Wersja B:



Bug czy Vulnerability ?

Zadanie!

1. Gdy nacisnę ALT+F9 program się “crashuje”.

2. Gdy wejdę na stronę www.przyklad.pl przeglądarka się “crashuje”.

3. Gdy oglądam zdjęcia z wakacji, przeglądarka obrazków się “crashuje”.

Bug czy Vulnerability ?

Zadanie!

BUG

1. Gdy nacisnę ALT+F9 program się “crashuje”.

VULN

2. Gdy wejdę na stronę www.przyklad.pl przeglądarka się “crashuje”.

VULN

3. Gdy oglądam zdjęcia z wakacji, przeglądarka obrazków się “crashuje”.

Najczęstsze Błędy

Najczęstsze Przyczyny Błędów

Buffer Overflow anyone?

Dokumentacja: Pakiet logowania:

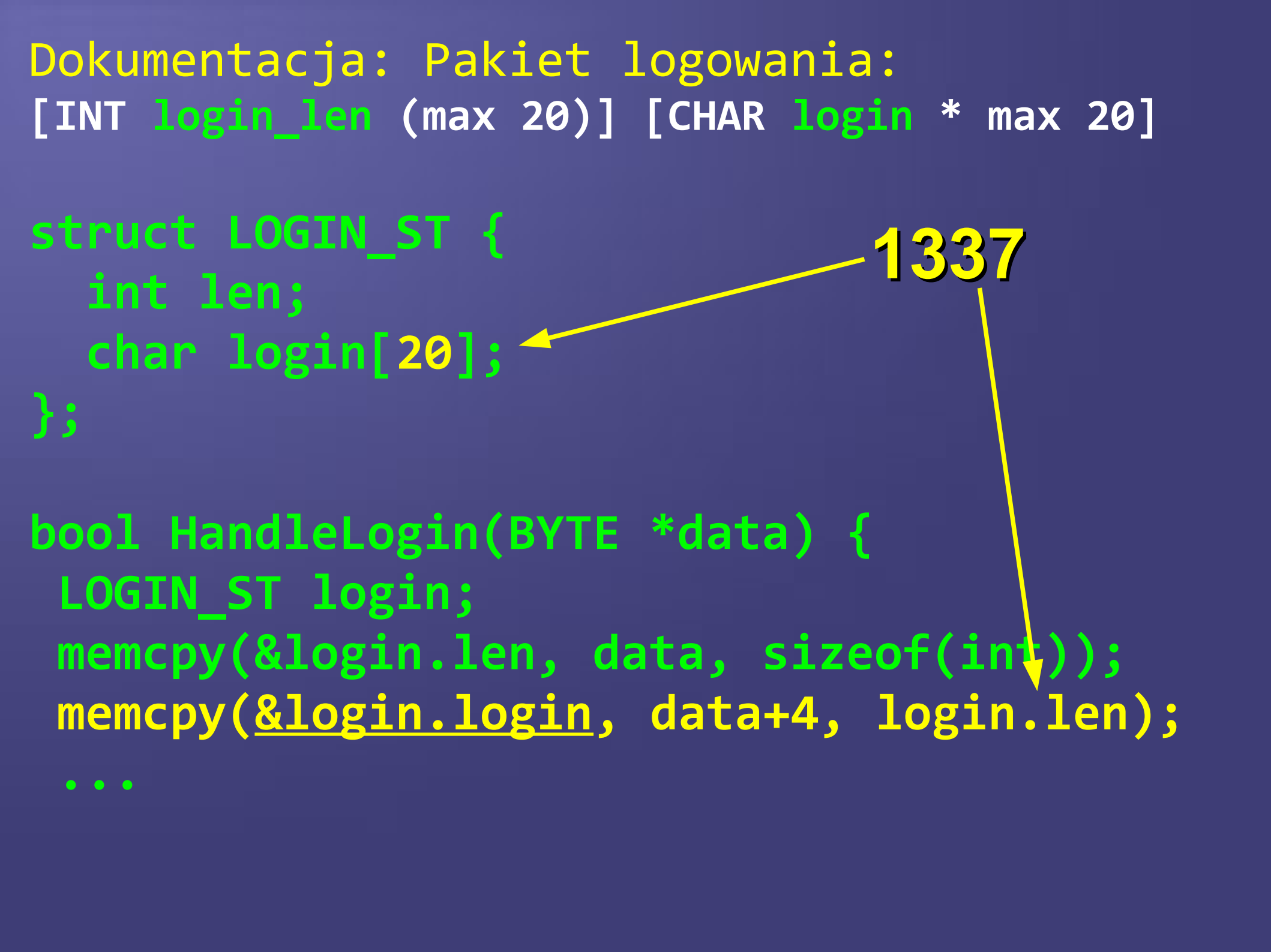
[INT login_len (max 20)] [CHAR login * max 20]

```
struct LOGIN_ST {
   int len;
   char login[20];
};

bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));
 memcpy(&login.login, data+4, login.len);
 ...
```

Dokumentacja: Pakiet logowania:

[INT login_len (max 20)] [CHAR login * max 20]

```
struct LOGIN_ST {
   int len;
   char login[20];
};

bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));
 memcpy(&login.login, data+4, login.len);
 ...
```



Przykładowy Skutek

Nadpisanie adresu powrotu znajdującego się na stosie.

aka remote arbitrary code execution

Przyczyna 1

Nadmierne ufanie dokumentacji.

“Ale w dokumentacji tak pisało...”

See also: ?id=179

Dokumentacja: Pakiet logowania:

[INT login_len (max 20)] [CHAR login * max 20]

```
struct LOGIN_ST {
   int len;
   char login[20];
};

bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));
 memcpy(&login.login, data+4, login.len);
 ...
```

Dokumentacja: Pakiet logowania:

[INT login_len (max 20)] [CHAR login * max 20]

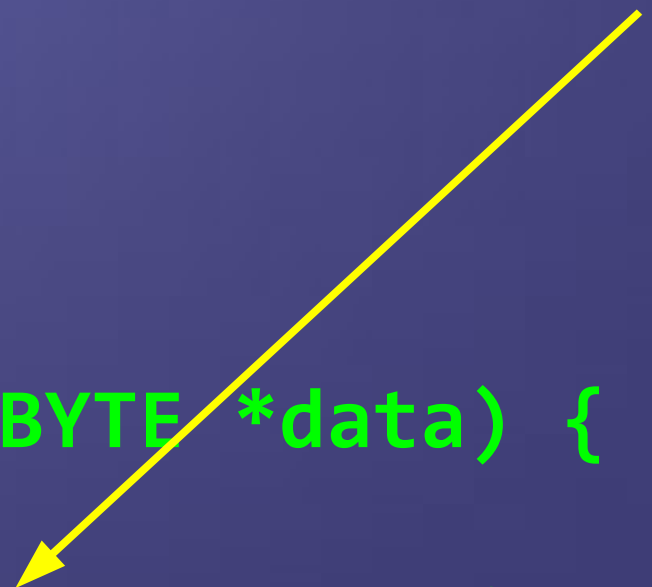
```
struct LOGIN_ST {
   int len;
   char login[20];
};

bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));
 if(login.len >= 20) throw “a fit”;
 memcpy(&login.login, data+4, login.len);
 ...
```

Dokumentacja: Pakiet logowania:

[INT login_len (max 20)] [CHAR login * max 20]

-1337

```
struct LOGIN_ST {
   int len;
   char login[20];
};

bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));
 if(login.len >= 20) throw “a fit”;
 memcpy(&login.login, data+4, login.len);
 ...
```



Dokumentacja: Pakiet logowania:

[INT login_len (max 20)] [CHAR login * max 20]

```
struct LOGIN_ST {
   int len;
   char login[20];
};

bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));
 if(login.len >= 20) throw “a fit”;
 memcpy(&login.login, data+4, login.len);
 ...
```

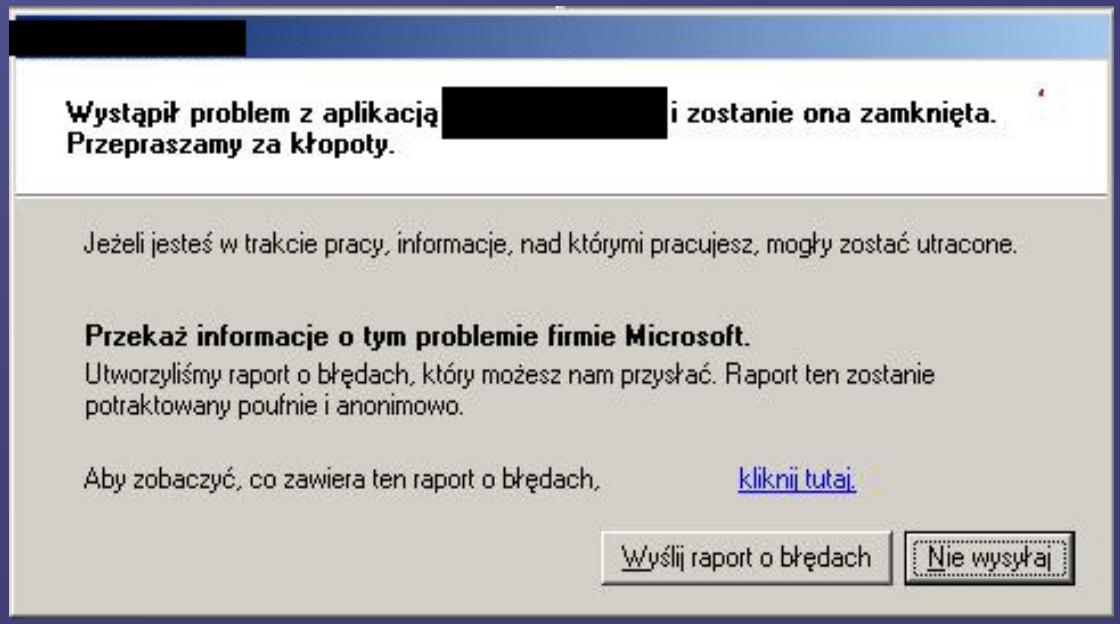
-1337

FFFFFAC7

4294965959

Przykładowy Skutek

“Program wykonał nieprawidłową operację I zostanie zamknięty...”



aka remote DoS

Przyczyna 2

Signed/Unsigned mismatch.

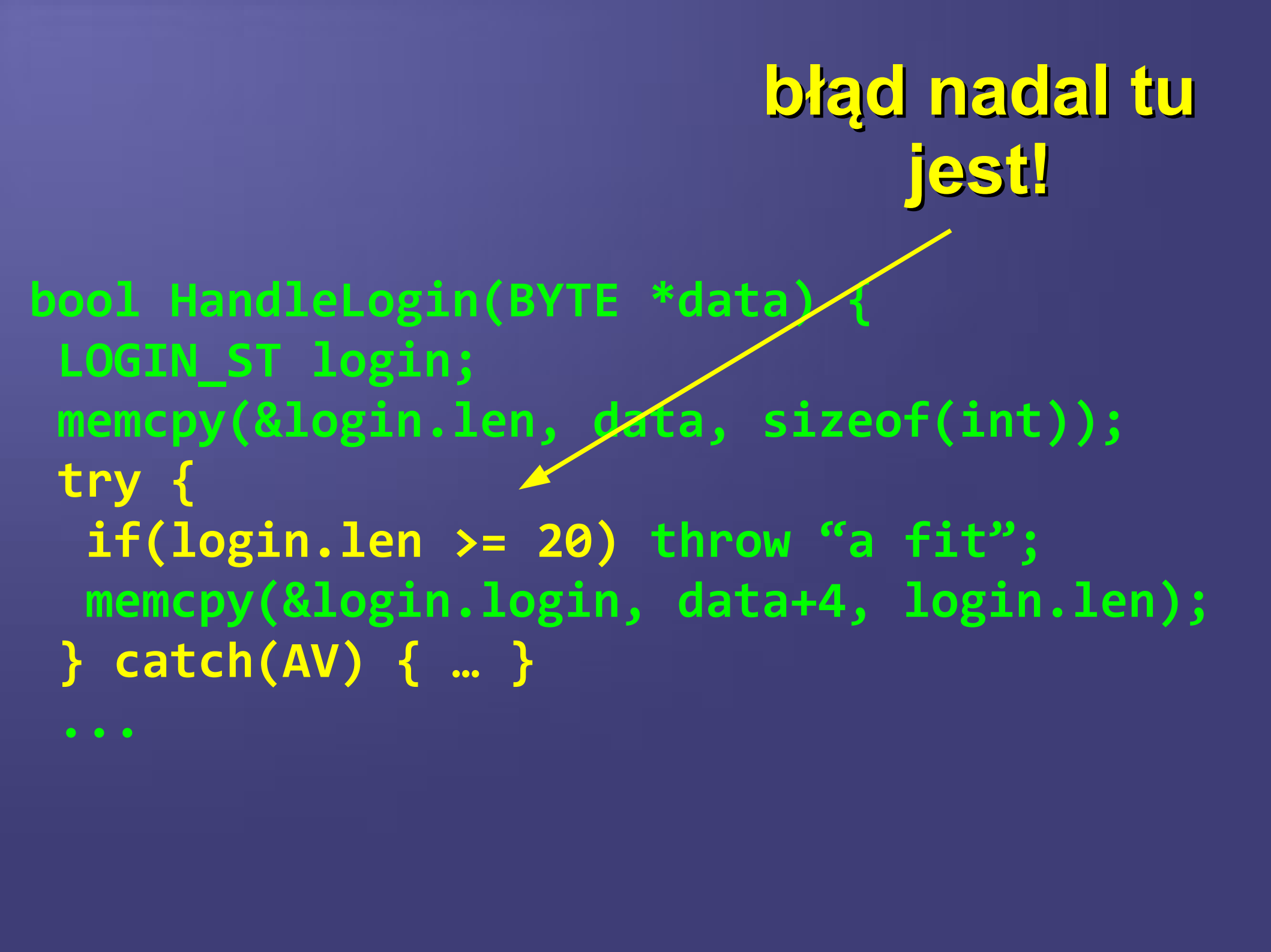
Wielkość nie może być ujemna.

Skoro rzucany jest wyjątek...

to go złapmy!

```
bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));
 try {
  if(login.len >= 20) throw “a fit”;
  memcpy(&login.login, data+4, login.len);
 } catch(AV) { … }
 ...
```

błąd nadal tu jest!



```
bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));
 try {
  if(login.len >= 20) throw “a fit”;
  memcpy(&login.login, data+4, login.len);
 } catch(AV) { … }
 ...
```

Przykładowy Skutek

Zostaje nadpisany adres SEH na stosie*

***nie dotyczy 64-bitów**

aka remote code execution... again

Przyczyna 3

Ukrywanie błędu.

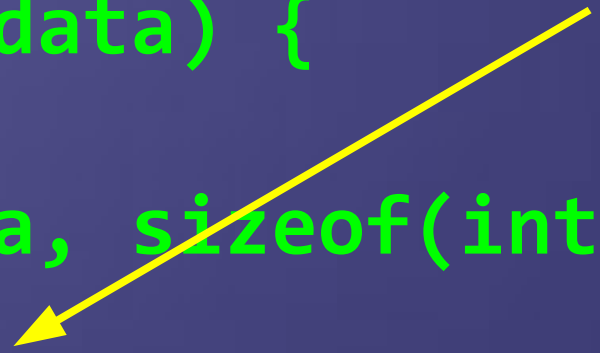
... zamiast jego poprawienia.

A może naprawmy ten pakiet...

```
bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));

 login.len = abs(login.len);

 if(login.len >= 20) throw “a fit”;
 memcpy(&login.login, data+4, login.len);

 ...
```

A może naprawmy ten pakiet...

abs(INT_MIN)=
INT_MIN
INT_MIN < 20 :)



```
bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));

 login.len = abs(login.len);

 if(login.len >= 20) throw “a fit”;
 memcpy(&login.login, data+4, login.len);

 ...
```

Przykładowy Skutek

Jeden z poprzednich...

aka remote code execution

or

remote DoS

Przyczyna 4

Próba naprawy uszkodzonego pakietu.

... zamiast jego odrzucenia.

```
struct LOGIN_ST {
  unsigned int len;
  char *login;
};

bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));

 // alloc +1 byte for \0 terminator
 login.login = (char*)malloc(login.len+1);
 if(!login.login) throw “a fit”;

 memcpy(login.login, data+4, login.len);
 ...
```

```
struct LOGIN_ST {
  unsigned int len;
  char *login;
};

bool HandleLogin(BYTE *data) {
 LOGIN_ST login;
 memcpy(&login.len, data, sizeof(int));

 // alloc +1 byte for \0 terminator
 login.login = (char*)malloc(login.len+1);
 if(!login.login) throw “a fit”;

 memcpy(login.login, data+4, login.len);
 ...
```

0xFFFFFFFF+1= 0
4294967295+1 = 0

Przykładowy Skutek

Jeden z poprzednich...

aka remote code execution

or

remote DoS

Przyczyna 6

mod 2**32

Niebezpieczne operacje arytmetyczne.

Wyliczanie ilości pamięci do alokacji wymaga używania funkcji które są “świadome” overflowów.
LUB
Zmiennych odpowiednich wielkości...

itd. & itp.

buffer overflow
buffer underflow
boundary condition error
race condition
time of check vs time of use
double free
use after free
integer overflow
integer underflow
format string bug
uninitialized variables
...

local privilege escalation
remote code execution
denial of service
information leak
...

XSS
HTML injection
SQL injection
XSRF aka CSRF
Local File Inclusion aka LFI
Remote File Inclusion aka RFI
“głębokie ukrycie”
mixed content
JSON hijacking
Clickjacking
Strokejacking
Cursorjacking :)
...

All this (and more!) is available in the programming language of your choice!

Tips & Trick

Nie ufaj “SHOULD”, “MUST”, “WILL”, etc w dokumentacjach.

Nie pisz własnych parserów - staraj się korzystać ze sprawdzonych bibliotek.

Naucz się myśleć jak atakujący.
Dowiedz się jak wyglądają “standardowe” błędy i jakie mogą być ich skutki.

Używaj fuzzerów do testowania kodu - to będzie pierwsze narzędzie które chwyci atakujący.

Nie ukrywaj błędów, poprawiaj je!

Warningi przy kompilacji po coś są :)

Nie używaj assert do “testów” związanych z bezpieczeństwem.
(assert'y istnieją tylko w kompilacji DEBUG)

gynvael.coldwind

Dziękuje za uwagę!

Czas na pytania :)

http://gynvael.coldwind.pl/
gynvael@coldwind.pl