ReverseCraft

by gynvael.coldwind//vx

008 – Użycie obcego kodu

Strony projektu:

http://re.coldwind.pl/
http://www.uw-team.org/

Obcy kod ?

kod w innym procesie / programie
który chcemy użyć (wywołać)

Obcy kod ?

kod w innym procesie / programie który chcemy użyć (wywołać)

np. rozbudowana procedura kryptograficzna (analiza malware)

np. wczytanie bitmapy w nieznanym formacie (game modding)

np. wywołanie procedury sprawdzającej hasło (brute force → crackme)

Metody użycia obcego kodu

program:
(cross-OS)

reassembling
inline
dynamic import

proces:

remote thread

Metody użycia obcego kodu

prawo autorskie
własność intelektualna

reassembling

(zwany również metodą Kopiego-Pasta)

co i jak?

1. disassemblujemy program źródłowy
2. wycinamy z niego interesujący nas fragment kodu
3. wklejamy do naszego programu (poprawiając składnie)
4. naprawiamy referencje

zalety:

prostota

wady:

czym więcej kodu, tym trudniej

różnice w składni trzeba korygować ręcznie

referencje

inline

(zwany również bezpośrednią metodą Kopiego-Pasta)

co i jak?

1. kopiujemy w hexedytorze kod maszynowy do programu
2. naprawiamy referencje

zalety:

bardzo szybkie wykonanie...

wady:

...ale tylko dla prostych przypadków

dynamic import
(prawie jak biblioteki)

co i jak?

1. ładujemy program do pamięci (LoadLibrary(reloc) / *read)
2. poprawiamy referencje w funkcji nas interesującej
3. wywołujemy funkcje

zalety:
prostota

wady:
referencje
czasami relokacje
protektory, bezpieczeństwo

remote thread
(manualne RPC)

co i jak?

1. uruchamiamy program źródłowy
2. przygotowujemy w nim środowisko
3. uruchamiamy w nim zdalny wątek
4. odczytujemy wynik

zalety:

nie trzeba walczyć z protektorami

trywialne dla funkcji o proto DWORD __stdcall X(PVOID)

wady:

większa złożoność problemu dla innych funkcji

bezpieczeństwo

Podsumowanie

Po pierwsze: znać metody
Po drugie: wiedzieć kiedy której użyć

Inne metody
np. emulacja funkcji

Dziękuje za uwagę :)

Strony projektu:

http://re.coldwind.pl/
http://www.uw-team.org/